(2)



# PROCESS ENGINEERING WITH THE EVOLUTIONARY SPIRAL PROCESS MODEL

**SPC-92079-CMC**

**VERSION 01.00.01**

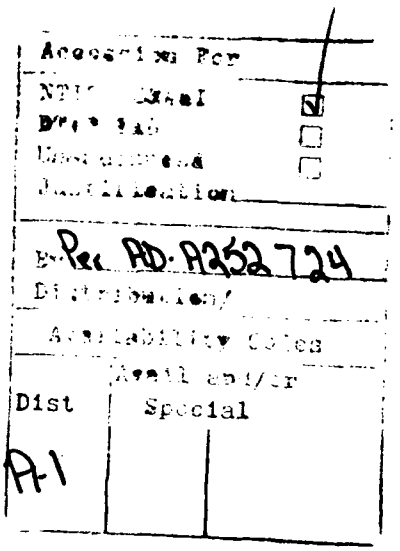
**DECEMBER 1992**





# PROCESS ENGINEERING WITH THE EVOLUTIONARY SPIRAL PROCESS MODEL






## SPC-92079-CMC

## VERSION 01.00.01

## DECEMBER 1992

Produced by the
SOFTWARE PRODUCTIVITY CONSORTIUM SERVICES CORPORATION
under contract to the
VIRGINIA CENTER OF EXCELLENCE
FOR SOFTWARE REUSE AND TECHNOLOGY TRANSFER

SPC Building
2214 Rock Hill Road
Herndon, Virginia 22070

Microsoft is a registered trademark of Microsoft Corporation.

Suntracs is a registered trademark of Sun Microsystems, Inc.

# CONTENTS

# FIGURES

# EXAMPLES

# TABLES

*This page intentionally left blank.*

# ACKNOWLEDGEMENTS

This technical report is based on, or makes reference to, several existing Consortium technologies.

- The Evolutionary Spiral Process (ESP) model technology was created and evolved by Kirsten Blakemore, John Blyskal, Ted Davis, Mary Eward, Robert Hofkin, Ph.D., Jim Marple, Tim Powell, Sue Rose, and Patricia Remacle.
- The ESP model technology was formally validated by Guy Cox, Ted Davis, Fred Hills, and Roger Williams.
- Synthesis technology was created by Grady Campbell, Jim O'Connor, Neil Burkhard, Jeff Facemire, and Rich McCabe.
- Software measurement guidance was authored by Robert Cruickshank, John Gaffney, and Richard Werling.
- Formal process modeling technology was created by Richard Bechtold and Robert Lai.
- Software engineering technology transfer guidance was authored by John Christian, Ph.D., Mary Eward, Sam Redwine, and Louis Tornatzky, Ph.D.

Kirsten Blakemore compiled, edited, and organized these technology sources into this technical report. Tim Powell developed the material for Appendix A. David Nettles provided overall guidance and oversight.

The review and comments of Grady Campbell, Mary Eward, Robert Hofkin, Ph.D., Curt Larock, Jim Marple, Sue Rose, Wil Spencer, Joseph Paul Valent, and Roger Williams greatly aided in the development of this report.

The coordination, editing, and processing of this document was performed by the Environment and Support Services Division.

*This page intentionally left blank.*

# 1. INTRODUCTION

> The first step is to accept the fact that change is a way of life, rather than an untoward and annoying exception.
>
> Frederick P. Brooks, Jr., *The Mythical Man-Month*

## 1.1 OVERVIEW

This report describes how the Evolutionary Spiral Process (ESP) model can be used to define and perform a *software development process* for a specific project. Although it is tempting to think that a software development process can be reused, without change, across projects within an organization, the reality facing most projects is that unique influencing characteristics, or *process drivers*, may result in software development processes which vary slightly to significantly across projects.

The ESP model is an adaptation of the basic *spiral model* proposed by Barry Boehm (Boehm 1986, 1988). The ESP model explicitly recognizes project management activities missing from traditional software development *process models*, and emphasizes *risk management*, or the ability to anticipate and respond quickly to faults such as problems or deviations from project plans. In addition, the ESP model supports evolutionary *process engineering* by allowing a project to engineer its process dynamically through defining, *instantiating*, *enacting*, and *evolving* its *process definition*. Because of these features, the ESP model provides the flexibility to identify and address project-specific process drivers continually, and to evolve the software development process accordingly.

## 1.2 SCOPE

The intent of this technical report is to explain how a project-specific software development process can be engineered using the ESP model. This report includes all of the general background and information necessary to cover the following topics:

- A general history of process, including an introduction to key process terms and concepts.
- ESP model concepts and principles.
- Evolutionary process engineering.
- ESP model-specific activity descriptions.

Because the industry has not yet set standards for concepts and definitions, it is recommended that the sections of this report be read in order. Otherwise, you may be confused by concepts presented, defined, or used in a way that is different from other sources.

## 1.3 ORGANIZATION

The following sections and appendices are included in this report:

- Section 1 describes the overview, scope, organization, and intended audience of the technical report.
- Section 2 presents an introduction to process, including: understanding process models; identifying and defining activities; the importance of process drivers; and process engineering.
- Section 3 explains the fundamental ESP model concepts of project management, risk management, and evolutionary development.
- Section 4 maps key process concepts to the ESP model to show how a project can engineer its project software development process in an evolutionary fashion by following the ESP model.
- Appendix A is a listing of example process drivers and desirable process characteristics.
- Appendix B presents the ESP activity specifications.

## 1.4 INTENDED AUDIENCE

This technical report is intended for use by the following audiences:

- A project manager interested in using the ESP model to plan, monitor, and control a project, and to define a software development process that best meets the needs and requirements of the project.
- A software engineering process group (SEPG) member, or other process improvement specialist, interested in defining an organizational process definition and tailoring it for use on a specific project.

## 1.5 TYPOGRAPHIC CONVENTIONS

This report uses the following typographic conventions:

Serif font ..... General presentation of information.

*Italicized serif* font ..... Words and expressions found in the Glossary and publication titles.

**Boldfaced serif** font ..... Section headings and emphasis.

# 2. INTRODUCTION TO PROCESS

> Process discipline provides the freedom for the most talented software professionals to be creative by freeing them from the many crises that others have created.
>
> Watts Humphrey, *Characterizing the Software Process: A Maturity Framework*

## 2.1 OVERVIEW

Imagine a standard, well-defined, and rigorously optimized process for constructing buildings. The activities in the construction process all have standard definitions, are always sequenced in the same way, and never vary in the time and *resources* needed to perform them. Type and quantity of material, as well as the number and skill mix of architects, engineers, and contractors, are known precisely. Standard tools and equipment are always used. Site descriptions, floor plans, blueprints, and other design and production documents never vary, in order to avoid changes to the standard process.

This construction process tends to produce square, concrete structures with the same dimensions, the same floor plan, and the same landscaping—well-built, reasonably priced, and quickly constructed, but of questionable value to customers who have unique requirements or constraints.

Differences in underlying culture, natural settings, and human-imposed expectations, decisions, and limitations are some of the reasons why construction processes differ from each other (Snyder and Catanese 1979). Such motivating factors, or *process drivers*, drive software development. Following a standard process for developing software may quickly produce high-quality and low-cost software, but the resulting software product(s) may not satisfy differences in organizational and client cultures; human-imposed objectives, alternatives, and constraints; and pre-existing knowledge or assets.

The industry is beginning to recognize the importance of standard and defined development processes to producing consistently high-quality software within budget and on schedule. However, unique project characteristics may impact a project's ability to perform to a previously defined development process: that is, different process drivers generally result in software development processes that may vary slightly to significantly from project to project.

This section provides an introduction to how process engineering helps to produce the right software development process for an organization and/or project. A software development process is the set of steps, or activities, for developing a software product and its supporting products; process engineering refers to the specific actions that attempt to ensure that a software development process is the best one, given organizational and/or project-unique process drivers.

## 2.2 PROCESS DEFINITION

A project's software development process can be physically represented by a process definition, or *artifact* documenting software development, management, and support activities. In order for a project

to actually perform, or enact, the software development process, the process definition should detail the following information:

- The set of activities necessary to develop software and its supporting products.
- How the activities relate to each other.
- A description of each activity.
- When to start and stop each activity.
- What inputs each activity needs.
- What each activity accomplishes or produces.
- Activity cost, schedule, and staffing estimates.
- What personnel, methods, practices, and tools will be used to perform each activity.

Essential key process activities included in a process definition can be represented at an abstract level through a process model when it is inappropriate to present detailed process definition information.

### 2.2.1 Process Models

During the 1950s and 1960s, software managers and engineers commonly operated in an environment where there were only three *general activities*: 1) talk to the programmer about what the *user* wanted, 2) wait while the programmer coded, and 3) repeat steps 1 and 2 until the final product was "fixed" enough that it approximated what the user wanted, and worked most of the time. This software development practice was like hiring a carpenter, giving him some wood and a hammer, talking to him about the house you have in mind, and then coming back in a few months to see what has been built.

Even today, many projects are developed in a "black box." High-level requirements disappear into a dark mass of frenetic activity. Although a product eventually emerges, and may even be the right one, no-one knows how it was produced, what decisions were made, and what, if anything, was learned to improve the next project. A black-box environment is not one where software development projects are planned or managed, but rather one that depends on the quality of the project team to produce the final product (Weber et al. 1991, 16).

The concept of process models emerged as a way to gain visibility into, and control over, the software development project. The definition of a software process model, adapted from Feiler and Humphrey (1992, 11), is an abstract representation of all or part of the process. The model is used when representing the complete process is not desirable or practical. Process models provide a consistent way to identify and order essential software development activities and to build in decision points for continuing to the next activity. Some well-known types of process models for software development and support are described below.

#### 2.2.1.1 Linear Development

The simplest process model is a linear sequence of development activities that presumes that there is never a need to go back to correct an error injected in a previous activity. The model presumes that

each activity can be completely, independently, and correctly performed before moving on to the next activity. Progress is measured in terms of the artifacts produced as a result of performing an activity, and the artifacts are used as input to the next activity. This view is often unrealistic for a project involving any degree of complexity, uncertainty, or possibility of change.

The *waterfall process model* (Royce 1970) is an extension of the linear sequence. The standard waterfall permits a project to back up stage-by-stage until it returns to the stage in which an error was introduced (Figure 1). The historical definition of the waterfall as a linear process restricts this back-up to corrective actions only. The model does not allow iteration for any other reason.

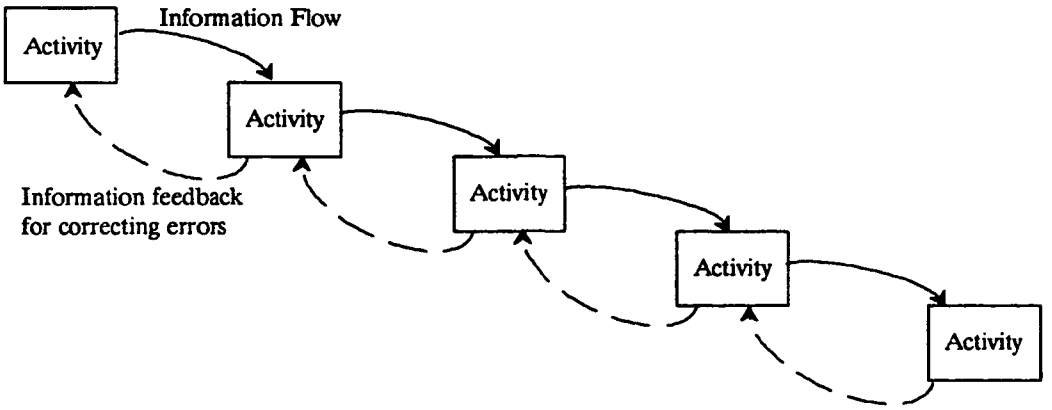



Figure 1. Waterfall Model

### 2.2.1.2 Evolutionary Development Model

The *evolutionary development* model, as described in Ould (1990) and illustrated in Figure 2, focuses on delivering several functioning versions, releases, or baselines of the software to the user instead of on the specific activities of software development. Rather than delivering just one version of the software, a sequence of functioning versions, each developed separately as a refinement of the preceding version, is delivered. With each delivery, user-identified changes are incorporated—such as those related to user interface, reliability, or performance—into the specification for the next delivery. The benefit is that the software continually evolves, based on user feedback, until that software is obsolete or needs to be replaced. At this point, the model prescribes that the process delivers a "final version" that will no longer be supported or evolved.

The development process for each version may follow the process prescribed by other models, such as waterfall or *spiral*. Regardless of the model used for each system's development, the emphasis is on delivering a system or prototype and allowing the user to gain experience with it. This experience is then fed back to the developer so that the definition of the next system can be improved in ways that have become important to the user, in addition to the usual corrections and extensions of the current system's code.

### 2.2.1.3 Spiral Model

The spiral model was originally developed by Boehm (1986, 1988) to address the known problems in more conventional, primarily waterfall, development models (see Figure 3). The spiral model attempts to provide a disciplined development structure, while eliminating the lock-step, linear progression of stages characteristic of the earlier models. It permits its users to avoid the automatic adoption of one-size-fits-all processes. The spiral is flexible and accommodates techniques such as prototyping, modeling, reuse, and automatic code generation.

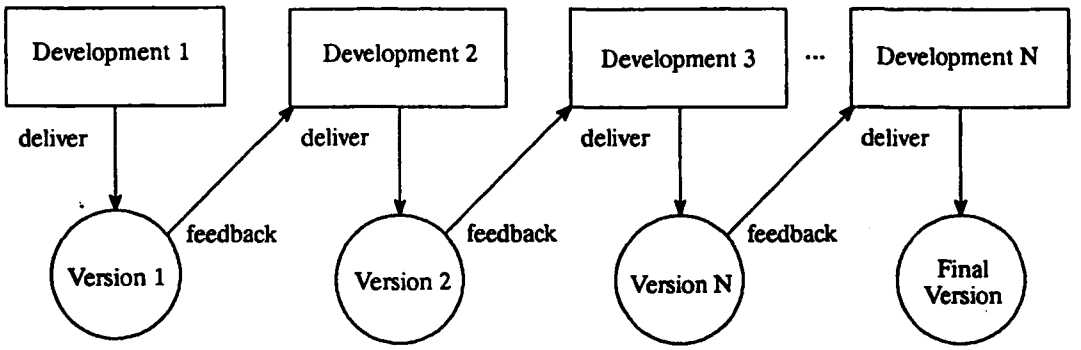



Figure 2. Evolutionary Development Model

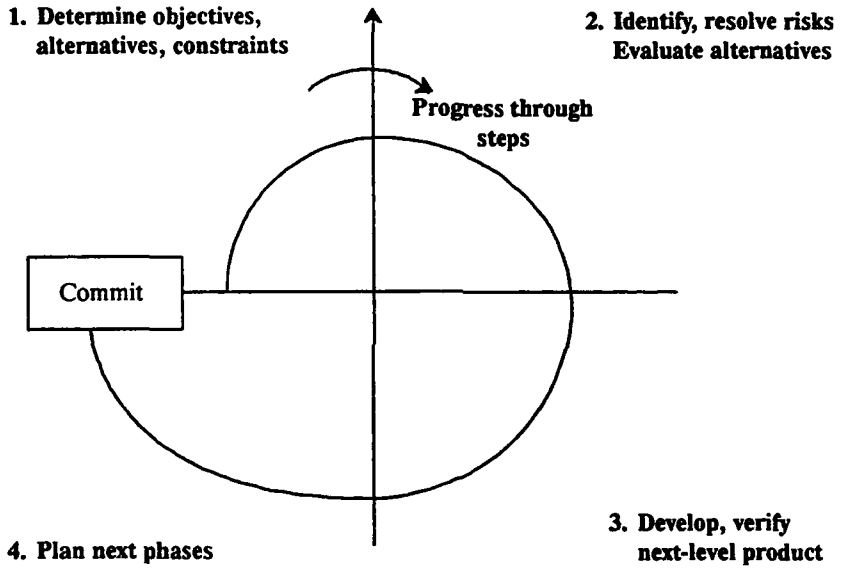



Figure 3. Boehm's Spiral Model

The spiral model is represented through four quadrants. Each of the quadrants represents a different range of activities, and a *cycle* traverses all four quadrants, represented by a 360° rotation in the graph that denotes that some aspect of the product has matured by a specified amount. Starting in the upper left quadrant (quadrant 1) determine objectives, alternatives, and constraints for the cycle, then move clockwise to risk analysis and aversion (quadrant 2), product development (quadrant 3), and planning and management (quadrant 4). The transition from the fourth quadrant to the first quadrant of the following cycle is controlled by committing resources explicitly to continue the project.

### 2.2.1.4 Synthesis Process Model

The *Synthesis process model* was developed by Software Productivity Consortium (1991) to support better use of expertise and knowledge about a set of similar problems and associated solutions within a specific *business area*, or group of projects supporting a similar product line or serving a similar customer base. Synthesis defines and integrates two processes: Application Engineering and Domain Engineering. Figure 4 shows how these two processes relate to each other.

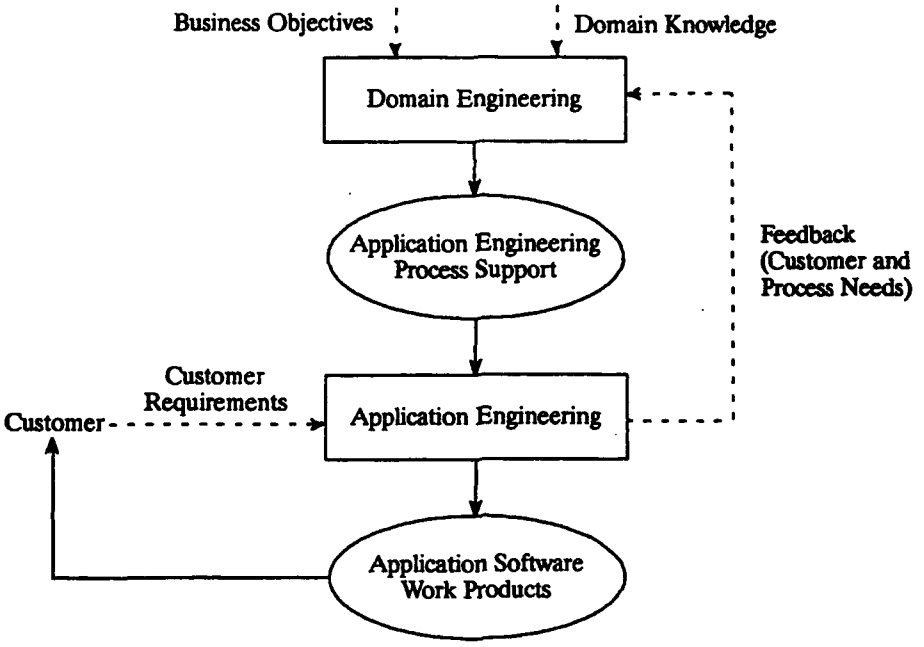



Figure 4. Synthesis Process Model

Application Engineering creates and validates a model for a required system. The model is a set of requirements and engineering decisions sufficient to describe a particular system. Work products, including code and documentation, are mechanically derived from this model using adaptable forms of those work products provided by Domain Engineering.

Domain Engineering supports Application Engineering in two ways. First, it creates a set of adaptable work products that correspond to those work products which an Application Engineering project must produce. Second, a standard Application Engineering process is designed that supports the decision making and work-product creation appropriate to projects in a business area.

### 2.2.2 Process Activities

As the discrete steps, or activities, of software development became visible through the use of process models, the concept of specifying those activities emerged in the industry as a way to standardize and control software development. Activities characterize what must be done to achieve or mature an objective or produce a product.

Activities can be specified informally or formally, and can be sequenced within the framework of a software process model. Once specified and sequenced, an activity can be considered enactable when the *methods*, practices, tools, and resources (dollars, time, equipment, personnel, etc.) to support the activity are determined, defined, and allocated.

Sources that offer software development activity and/or artifact specifications include DOD-STD-2167A, IEEE STD 1074, and ISO-9000. Other actions that support software development, such as reuse identification, classification, and adoption activities, are identified and specified in various guidebooks and textbooks. These sources provide a useful "starter set" for specifying activities. They can be adopted verbatim or adapted to reflect an organization's existing infrastructure, culture, language, policies, procedures, standards, and other process drivers.

Activities are further supported by methods, or a systematic set of tasks, rules, guidelines, and techniques that describe how to perform an activity. Many methods are supported by tools. For example, the activity of developing a project schedule is supported by several scheduling methods, such as Gantt and Project Evaluation and Review Technique (PERT). Both of these methods are supported by a variety of automated tools such as Microsoft Project or Suntracs.

Activity inputs and outputs are physically represented through artifacts, such as a product development plan or a detailed design document: that is, an artifact is the product produced by an activity or series of activities within a process. The output artifact, or part of the artifact, of one activity is often used as an input to one or more subsequent activities. Artifacts are frequently subject to specific templates, such as the DOD-STD-2167A Data Item Descriptions (DIDs). A degree of activity sequencing is implied when using activity artifacts as input to another activity. If the detailed design document is a necessary input to coding activity(ies), then the activity produces the detailed design document should be performed sometime before the coding activity, though not necessarily immediately before.

*Notations* can used to specify activities in a consistent way. In general, notations provide the format of an activity description, including activity inputs, outputs, and start and completion criteria. A notation can be an informal description of an activity using natural language and standard formats, or a formal specification language. The more informal notations are sufficient for practitioner's process manuals; the more formal notations lend themselves to *process automation*, or the use of a machine to automate all or part of the software development process. Some notations can be adapted to specify artifacts in terms of what activity(ies) generate the artifact as an output or use the artifact as an input. Two moderately formal notations are briefly described in the subsections below.

#### 2.2.2.1 Entry-Task-Validation-Exit Notation

Figure 5 is a graphic depiction of an activity based on the entry-task-validation-exit (ETVX) paradigm (Radice and Phillips 1988). The entry and *exit criteria* restrict products from entering or leaving the activity unless they are at a prescribed state of completeness. The *task(s)* performs the real work of the activity. *Validation* tracks the quality and completeness of the activity.

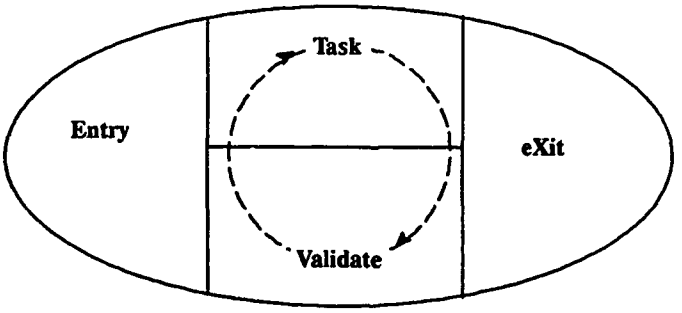



Figure 5. Entry-Task-Validation-Exit Notation

An activity can be performed as soon as all its *entrance criteria* are met. These entrance criteria should include a check of the state of its inputs. The typical operation of an activity defined using this notation is to transform input products into the next level of completeness by performing a task. A task is repeated as necessary until the validation shows that the task is satisfactorily completed and the exit criteria are met. The ESP activities provided in Appendix B are specified using a modified ETVX notation.

### 2.2.2.2 Structured Analysis and Design Technique Diagrams

Structured Analysis and Design Technique (SADT) diagrams (Marca and McGowan 1987) make the relationships among activities more evident. This notation represents an activity as a box whose left side shows inputs, right side indicates outputs, top indicates control inputs, and bottom shows the mechanisms, or method, by which the input will be transformed into the output. Figure 6 presents an example of SADT notation. Arrows connect boxes and represent information flow. Normal text on an arrow represents a data item; italics means a condition. Controls are inputs that the activity uses but does not transform.

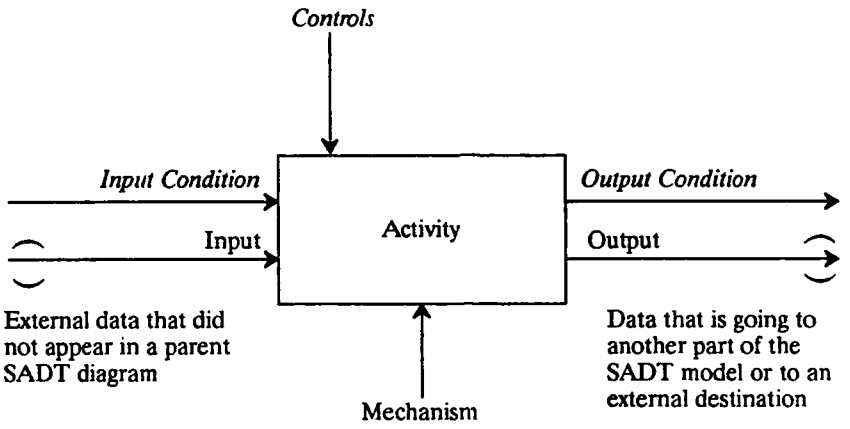



Figure 6. Structure Analysis and Design Technique Notation

The SADT notation is used throughout Section 3 to help illustrate ESP concepts and principles.

## 2.3 PROCESS ENGINEERING

The intent of process engineering is to define and evolve a development process definition that best addresses an organization's and/or project's unique process drivers. Figure 7 represents one view of how to engineer a software development process.

Specific software development processes may be defined differently for different organizations or for different projects within an organization. Activities may be identified, specified, represented, resourced, and supported differently, depending on unique motivating factors or process drivers. A process definition does not need to be engineered from scratch for each organization or project; parts of a software development process(es) can be standardized and reused.

At the organizational level, process engineering can start by identifying existing process models, activity specifications, method descriptions, and other process information, material, and/or definitions, and using them as the basis for creating an organizational process definition or the standard software development process at an organizational level. Parts of the process definition that may be unique to the organization, or where no material exists or is appropriate, might need to be created from scratch.

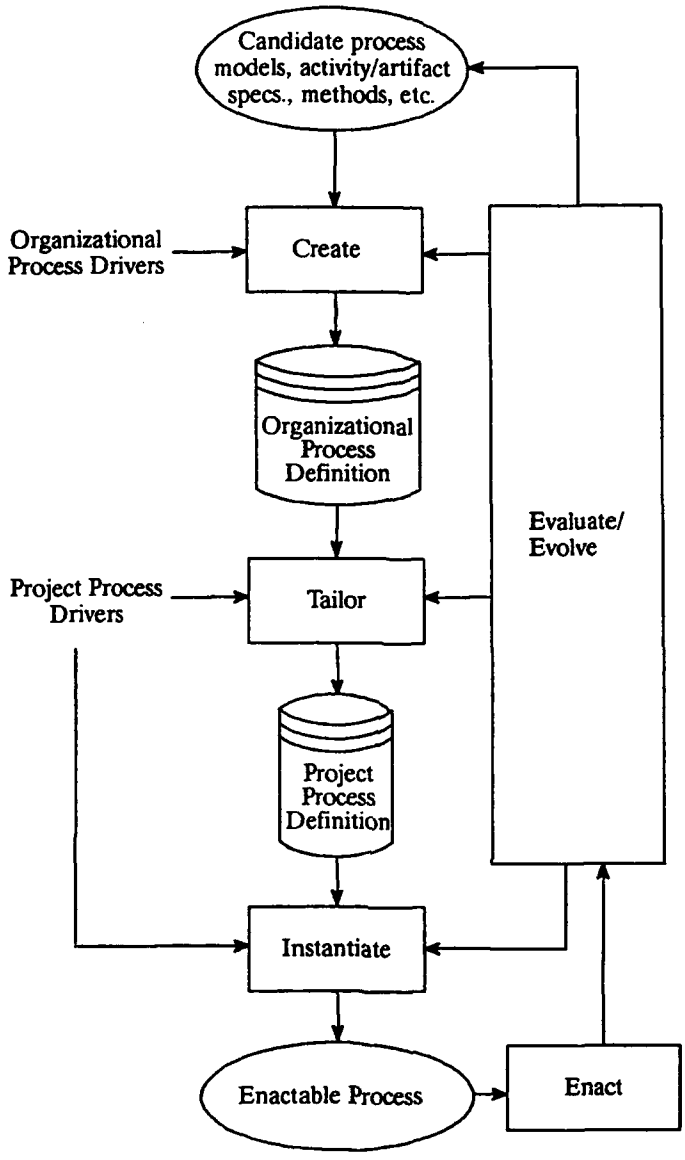



Figure 7. A Process Engineering Model

At the project level, an organizational process definition may need to be tailored to a project process definition, based on a project's process drivers. As the project progresses, the project process definition is instantiated, or detailed to an enactable level. As a result of enacting the process, the project process definition may be evaluated and evolved to address new or better-understood process drivers, correct problems, and/or incorporate lessons learned. Continuous *process evaluation* may also include activities that analyze process measurements and recommend process improvement, evolution, and automation at project and organizational levels.

### 2.3.1 The Impact of Process Drivers

Process drivers are key characteristics that directly affect the definition of the software development process at the organizational level, as well as at the level of an individual project within that organization.

Boehm and Belz (1988) discuss the impact of process drivers when selecting a process model; process drivers also impact process definition, instantiation, and evolution.

Before a process definition can be developed for an organization or project, process drivers should be identified and analyzed. For example, an organization may have recently invested in training, tools, and support for an object-oriented design method. As a result, process definitions at the organizational or project level are likely to reflect this investment by specifying object-oriented design activities and describing any supporting tools. Appendix A provides a partial listing of process drivers.

#### 2.3.1.1 Organizational Process Drivers

It is possible that several parts of a standard software development process can be reused to some degree by projects within an organization. However, a standard process definition is most likely to vary across organizations, based on unique process drivers. *Organizational process drivers* are the key characteristics of a specific organization, such as a corporation or division, that impact an organizational process definition. Process drivers that may affect the identification and definition of activities at an organizational level include:

- Investments (such as equipment or methods in which the company has invested).
- Technology and business strategies.
- Infrastructure and culture.
- Government rules and regulations.
- Market trends and business goals.
- Policies, procedures, and standards.

For example, a commercial company has certain organizational infrastructure and investment characteristics that directly impact the testing activities of the software development process for all projects. Specifically, this commercial company requires that its well-staffed and seasoned, independent test department perform rigorous testing and evaluation activities on all software prior to final production and release. Another commercial company does not maintain an independent test staff, but has expanded its market-oriented culture to include fielding beta versions of the software to preselected individuals in the market place as the mechanism for final testing and evaluation. As a result, the software development processes for the projects within each company will be defined differently to reflect the two approaches to final test and evaluation.

#### 2.3.1.2 Project Process Drivers

*Project process drivers* are the key characteristics that impact process definition at the project level. Project process drivers may vary in type, priority, and impact, depending on the specific project. Process drivers that may affect process definition for a specific project include:

- The organizational process definition, or organizational process drivers, if no organizational process definition exists.

- Project objectives, such as winning a follow-on contract or producing zero-defect software.
- Project *constraints*, such as customer requirements or environment.
- Project *risks*, or potential problem areas.

For example, one project is driven by cost constraints. It therefore selects inspections as a verification method for all requirements, design, and coding activities since a team member has experience with the method and can inexpensively train the rest of the team. Another project is driven by a schedule that does not allow for full-scale inspections. Therefore, it uses inspections to support the preliminary requirements and preliminary design activities, and informal walkthroughs to support the other requirements, design, and coding activities. One project spends months on the preliminary requirements activity to identify accurately high-level requirements with the customer. Another project is producing the third version of the software and spends only a few weeks on the preliminary requirements activity since requirements were defined and refined during development of the previous software releases.

### 2.3.2 Define an Organizational Software Development Process

The Capability Maturity Model (CMM) developed by the Software Engineering Institute (SEI) at Carnegie Mellon University attempts to assess and characterize an organizational software process in terms of one of five levels of process maturity. The five levels of process maturity are described in Paulk et al. (1991). They range from an ad hoc and immature process, characterized as a level one, up to an optimizing and continuously improving process, characterized as a level five. Once an organization is assessed, it can strive to improve its process maturity level as guided by the key process areas (KPAs) of the CMM. KPAs are the key process areas that an organization must address satisfactorily before progressing from one maturity level to the next. Defining an organizational software development process is one of the KPAs an organization must address when trying to move from level two, or repeatable process maturity, to level three, or defined process maturity.

The first step for creating an organizational process definition is determining what constitutes an organization. One company may feel that it has several divisions, each with sufficiently different missions, customers, products, and cultures to warrant creating a separate organizational process definition for each division. Another company may determine that, given its size, more than one organizational process definition would not make sense.

As discussed in Section 2.2, many materials and artifacts are available in the industry that can be used as a starting point for creating an organizational process definition. An organization's process improvement group may be the best choice for performing organizational process definition activities such as:

- Identifying and analyzing organizational process drivers.
- Developing an organizational process model.
- Identifying and adapting existing process materials and artifacts based on organizational process drivers.
- Defining and continually improving the process definition at an organizational level.

- Achieving consensus on the organization process definition by all *stakeholders* or interested parties.
- Training individuals within the organization on the defined process.

The level of detail to which an organizational process definition is documented varies from organization to organization. Activity scope, the cost, schedule, and staffing to support an activity, and the specific personnel, methods, practices, and tools used to perform each activity, however, are process drivers that generally vary from project to project. As a result, it may not be effective to document an organizational process definition to an enactable level of detail.

The internal process guidebook appears to be the most common mechanism for documenting an organization's software development process. These internal guidebooks are generally presented in the form of policies and procedures, and are intended for use as the foundation for more definitions of detailed project process. In addition to documenting and refining the internal process guidebook, organizations at higher levels of process maturity collect and analyze quality, size, cost, schedule, and other measures of the software process and product, as well as automate appropriate parts of the process.

When developing an internal process guidebook, an organization can use the *Process Definition and Modeling Guidebook* (Software Productivity Consortium 1992a). This book provides a *tailorable* approach for formal process definition and modeling. Specifically, the guidebook provides a flexible set of templates and techniques for capturing and representing process in terms of the relationships and constraints between and within activities, artifacts, and supporting resources. The techniques documented in the guidebook attempt to represent the process in terms of an optimal combination of text and graphics.

### 2.3.3 Defining a Project Software Development Process

Identification and analysis of project process drivers generally results in process tailoring requirements and justification. For example, a project may need to tailor the life-cycle activity descriptions documented in the organizational process definition because of an external customer constraint to comply with DOD-STD-2167A. Based on its unique process drivers, a project will require the flexibility to tailor the organizational process definition to a project-level process definition by:

- Determining the project process model by adding to or deleting from the set of activities included in the organizational process definition and reorganizing activity sequences and relationships.
- Rewriting activity specifications.
- Adapting or changing what each activity accomplishes or produces.

A project can define and instantiate its entire project process definition either before it is enacted, or in parts, as long as each part is defined and instantiated prior to enactment. Specifically, a project can instantiate its project process definition by:

- Receiving consensus on the project process definition by all stakeholders.
- Allocating costs, schedules, and responsible staff for each activity.
- Determining and committing methods, practices, and tools for each activity.

It is possible to use a project's software development plan as the mechanism for documenting a high-level project process definition. The schedule usually included in a project plan can also be used to represent the process in terms of the set of activities essential to the project and their sequences and relationships, inputs and outputs, start and end dates, responsible staff, and references to any known methods, practices, or tools selected to support the activity. Other project information such as a *work breakdown structure* (WBS), detailed schedules, activity specifications, and method, practice, and/or tool descriptions may be either included as an appendix to the software development plan or documented separately.

A complete project plan can define a project's process. Because of this, it is possible for projects that adequately perform basic planning and management to define and enact a reasonable software development process in terms of activity selection, description, sequence, cost, schedules, and responsible staff. It may be difficult, however, for an individual project to support detailed activity specification activities and the identification, purchase, and transfer of the best state of the practice methods, practices, and tools. In addition, it may be very difficult to attain higher levels of process maturity when working from project processes that are constantly moving or changing in ad hoc, uncontrolled ways. Process reuse, measurement, and continuous improvement can be achieved more easily when a standard organizational process definition is developed, based on project process drivers, and used as the basis for tailoring and evolution.

### 2.3.4 Perform Continuous Process Evaluation

Process improvement stems from using the process and evaluating how well it works, and from evaluating and monitoring the software product being produced as a result of enacting the software development process. Process evaluation techniques can include:

- Internal process assessments.
- Ongoing measurements and metrics.
- Post mortem evaluation.

An internal process assessment assists an organization to determine and characterize its internal process maturity. The purpose of a process assessment is two-fold. First, a process assessment provides an organization with a characterization of current process maturity and guidance on the key activities necessary for evolving to higher levels of process maturity. Second, a measure of organizational process maturity can be used as a way for a customer to assess the software development capability of potential contractors or vendors. An organization interested in assessing and improving its internal process can use the assessment technique and materials developed by the SEI, and based on the CMM discussed in Section 2.3.2

Measurements that a project might collect as an ongoing part of evaluating and monitoring product development include traditional cost, schedule, and size measures. In addition, a project could identify and classify errors discovered and solved, or the type and severity of risks identified and mitigated, when performing each activity of the process. While cost, schedule, size, error, and risk measurements are all very useful project management tools, they can also be used to build process metrics, such as the average percentage of cost and time spent at each activity, or the identification of the activities where errors and risks are typically discovered or introduced.

A post mortem evaluation of a project's process definition and enactment might also identify potential reusable process assets such as process(es), methods, tools, and measurements, as well as the rules for reusing these process assets. Process evaluation can include classifying the different software processes into families of similar processes. Although unproven, process metrics of cost, time, errors, and risks may be more accurate when analyzed in terms of business areas.

In addition, a post mortem evaluation can analyze lessons learned and determine potential process improvements that may impact the organizational process definition, or process automation opportunities to support enactment of an organizational process definition. If projects misinterpret, constantly tailor, or simply bypass a certain part of the organizational process definition, then the organizational process definition may need to be evaluated and modified.

## 2.4 SUMMARY

This section introduced some of the basic process concepts, including:

- ***Software Development Process.*** The set of steps, or activities, for developing software and supporting products.

- ***Process Definition.*** A physical product that defines a software development process. A process definition can be represented at an abstract level by a process model or may include all the detailed information necessary to enact the process. An organizational process definition describes a standard software development process at an organizational level and is often documented in terms of global policies and procedures in an internal guidebook. A project process definition is a specific tailoring of the organizational process definition that best addresses project's needs and can be documented as an integral part of the project plan. A project process definition should detail all of the information necessary to actually enact the process; an organizational process definition may not need to be enactable.

- ***Process Engineering.*** The specific actions that attempt to ensure that a process definition is the best one, given an organization's or project's unique process drivers. Process engineering includes identifying and analyzing organizational process drivers; developing an organizational process definition; identifying project process drivers; tailoring an organizational process definition to a project process definition; evaluating and evolving a project process definition as a project progresses; and evaluating the final process definition and enactment to identify trends, statistics, process improvements, and process automation to support enactment of a process definition.

Section 3 of this report discusses the principles and concepts behind the ESP model. Section 4 of this report combines the introduction to process presented in this section with the ESP model principles and concepts to discuss how a project can engineer its process using the ESP model.

*This page intentionally left blank.*

THIS
PAGE
IS
MISSING
IN
ORIGINAL
DOCUMENT

17 & 18

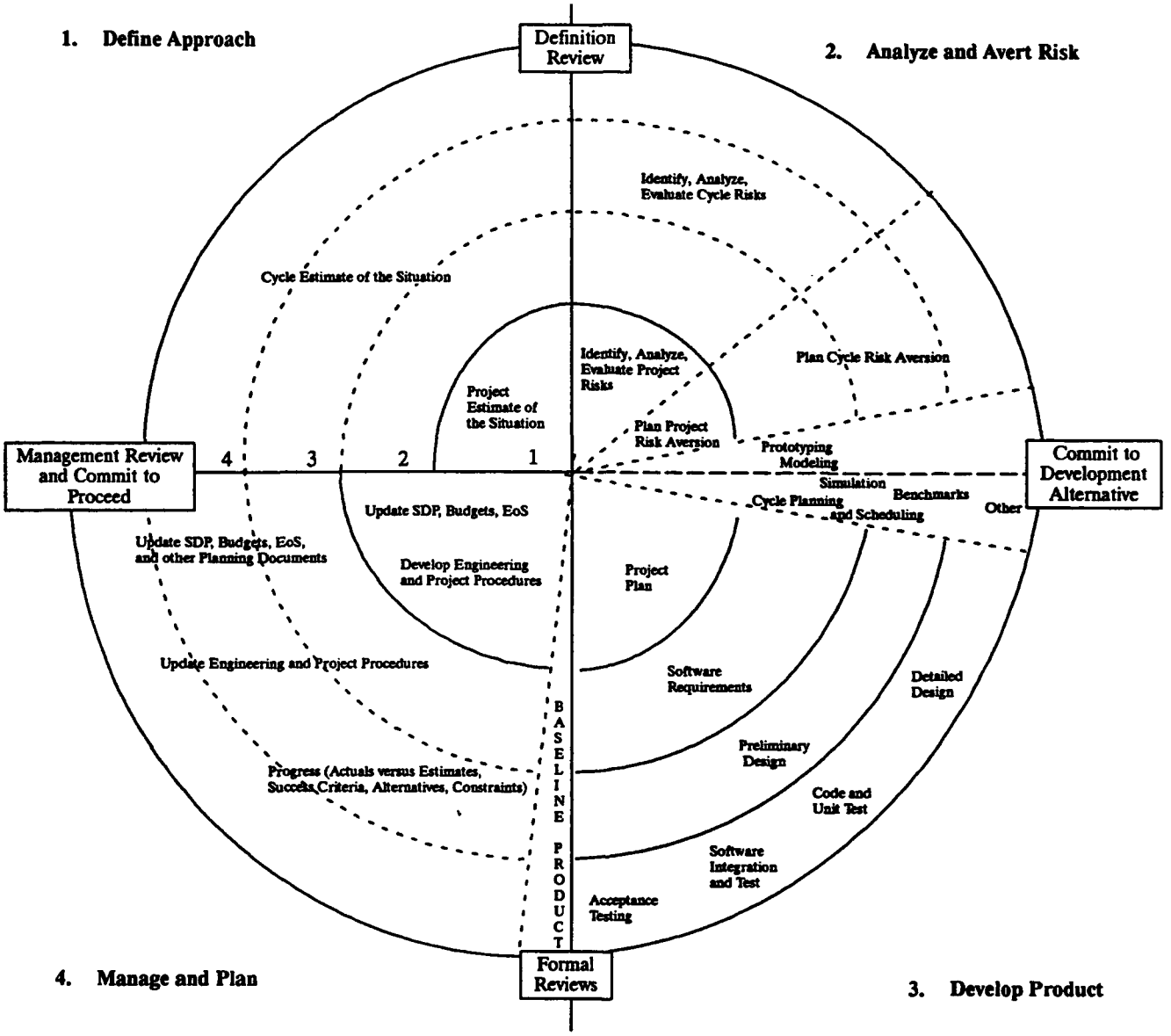



Figure 9. An Example Spiral

The subsections below discuss how a project traverses each of the four quadrants and performs the activities within each quadrant.

### 3.2.1 The First Quadrant: Define Approach

The approach definition establishes the ground rules for measuring progress either at a high-level for the project, or at a more detailed level for the current cycle.

During the first cycle, the project develops an *Estimate of the Situation* (EoS), which documents project-level process drivers, including objectives, success criteria, alternatives, constraints, internal and external factors, and other information. The definition review at the end of the first quadrant authenticates and establishes consensus on the project-level information contained within the EoS.

During subsequent cycles, the project updates the EoS to include cycle-specific objectives, *success criteria*, alternatives, constraints, internal and external factors, and other information that will drive the process for that specific cycle. The definition review serves to authenticate and establish consensus on the cycle-specific information.

The specific concepts and activities of the first quadrant are graphically depicted, using the SADT notation described in Section 2.2.2.2, in Figure 10 and discussed further in the subsections below.

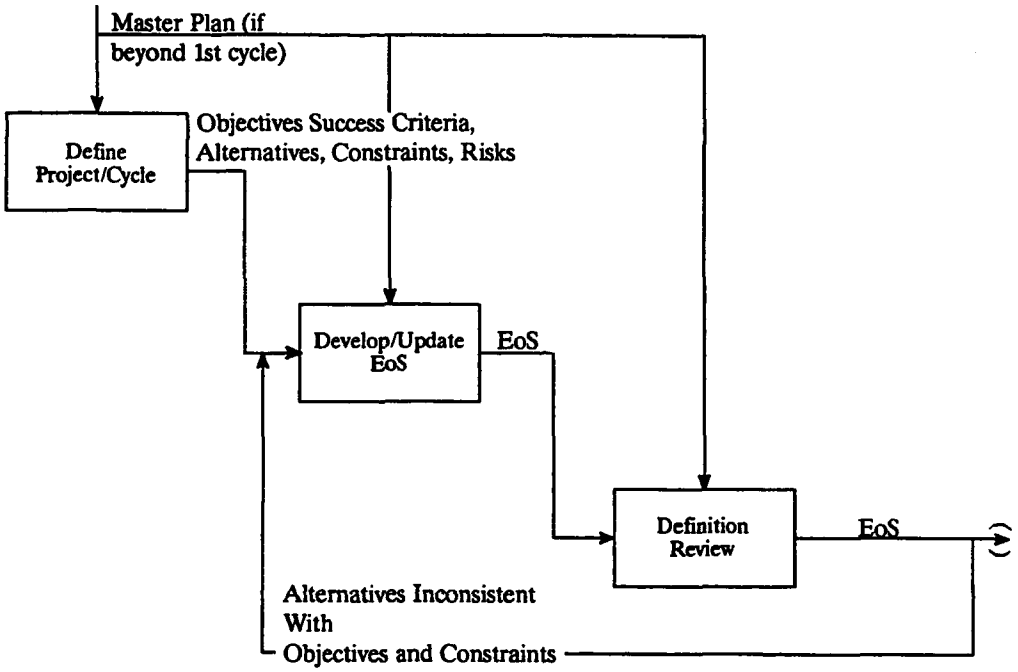



Figure 10. Define Approach Activities

### 3.2.1.1 Define Project/Cycle

A project must define the following characteristics. The definitions must be precise enough to establish objective measures to determine if the project and/or cycle is meeting expectations, or if the expectations are infeasible:

- ***Objectives.*** What should be accomplished during the project and/or current cycle.
- ***Success Criteria.*** How to know when the project and/or current cycle is completed.
- ***Alternatives.*** Different ways to meet the objectives.
- ***Constraints.*** Limitations on alternatives.

#### 3.2.1.1.1 Objectives

An objective is the intended or desired result of a course of action. Within the context of the ESP model, a project should state its objectives at two levels of abstraction: as high-level project intentions as documented in the *Master Plan* produced during the first cycle of the spiral (described in Section 3.2.3.3)

and as specific intentions the project should accomplish during current cycle. Objectives should be appropriate to the current level of project abstraction, realistic, and measurable.

These points, from Odiorne (1979), may be useful when defining objectives:

- A mistake in objectives will produce a mistake in an activity.
- Every objective that is accepted means that some other objective has been rejected.
- Set lower objectives for low confidence (or high risk); set ambitious objectives for high confidence.
- Force a measurement; if it does not work out, drop it.
- Side effects cannot always be predicted in detail.
- Experience is a pretty good teacher, and experienced objective setters set better objectives.

It is better to write several independent objectives than to pack everything into one statement; progress is assessed more easily when objectives are separated. Isolated objectives also simplify tradeoff analysis. A rule of thumb is that each objective may be met independently of the success or failure of any other objective for the cycle. It may be helpful to organize project objectives into a hierarchy, which can then be expanded into cycle objectives.

A good objective has several characteristics. It is:

- ***Results Oriented.*** The objective describes the desired result, not the means to achieve it.
- ***Measurable.*** It is possible to track how closely and how well the objective is met.
- ***Clear and Concise.*** The objective can be stated in 20 or fewer words.
- ***Controllable.*** It is possible to influence how, whether, and to what degree the objective is met.
- ***Realistic.*** It is possible to achieve the objective, given current information and resources.
- ***Appropriate.*** The objective is well suited to the current level of abstraction.

#### 3.2.1.1.2 Success Criteria

Where possible, a project should quantify its objectives so that there is an unambiguous measurement of completion. If objectives are expressed in a quantifiable way, they will automatically form the basis for the success criteria, which represent the minimum that must be accomplished in the cycle so the project can make progress.

The Department of Defense (DoD) now recognizes a success criterion (or minimum acceptable requirement) as "the value for a performance parameter . . . necessary to provide an operational capability that will satisfy the mission need" (Department of Defense 1991, 4-B-1). For example, an objective might be throughput of 100 transactions per minute for a particular system, but the success criterion could be 80 transactions per minute. Although the cycle will try to meet its objectives, it can still terminate when the success criteria are met.

#### 3.2.1.1.3 Alternatives

The objectives look at the **things** that must be done; the alternatives address the **ways** that the objectives might be achieved. The project should document the following information about each alternative:

- An outline of the activities and/or methods needed to accomplish the objectives.
- Any interdependencies among the activities/methods.
- An estimate of the resources (e.g., people, information, time, materials, equipment) necessary to complete each activity/methods.
- An estimate of the confidence (lack of risk) in the estimates.
- An estimate of any additional benefits and/or opportunities.

At this time, a project should include any reasonable alternative, even though some strategies may appear to be better than others. Comparisons among the alternatives are made in the second quadrant.

#### 3.2.1.1.4 Constraints

Constraints are unchangeable considerations that any alternative must satisfy. All viable alternatives must conform to the project constraints. Alternatives selected by a project in earlier cycles tend to impose constraints on later cycles. Project constraints may include:

- ***External Influences.*** The project is not autonomous. It is part of a larger business area organization that imposes expectations and constraints on the project (e.g., corporate goals and existing process assets such as activity definitions, predetermined methods, policies, procedures, or standards). The customer also imposes expectations and requirements on the project.
- ***Limited Resources.*** The project is limited in its skilled personnel, the time to develop the system, the funding available to support the project, and/or the development environment support.
- ***Existing Technology.*** The project is tied to existing technology (such as reusable parts; commercial off-the-shelf (COTS) technology; and software development methods).
- ***Standards.*** The project's parent organization or the customer imposes development standards to define sets of activities to be performed. The *IEEE Standard for Developing Software Life Cycle Processes* IEEE STD 1974 (IEEE Computer Society 1992) and the *Military Standard: Defense System Software Development* DOD-STD-2167A (Department of Defense 1988) are two of the more well-known industry standards.

The project should evaluate constraints for how unchangeable they really are. Self-imposed and arbitrary constraints are common, and they make alternatives unnecessarily difficult. For example, some factors, such as choice of development environment, that at first may appear to be constraints could in fact be assumptions about the alternatives, or may constitute a requirement that a customer may be willing to change.

#### 3.2.1.1.5 Develop/Update Estimate of the Situation

A project creates the EoS during the first cycle of every spiral. At a minimum, the EoS should identify, analyze, and document the project's process drivers in the form of objectives, alternatives, product

and process assumptions, and the assets available for performing the project (Charette 1989). The project can identify process drivers from sources such as the contract and Statement of Work; insights about and from the client; historical project plans and budgets; client and business area policies, procedures, and standards; and interviews with key project personnel at all levels. Example 1 is an EoS template adapted from Charette (1989).

Introduction
- Overview
- Revision History
- References

Project Description
- Project Mission
- Project History

Project Characteristics
- Project Objectives
- Assumptions
- Constraints

Project Assessment
- Assessment of Factors Influencing Success
  - Internal Factors
    - Organizational Factors
    - Process Factors
    - Technical Factors
  - External Factors
    - Customer Interface
    - User Interface
- Assessment of Requirements
  - Schedule
  - Cost
  - Product Capability
  - Quality

Possible Courses of Action

Conclusions

Reviewers

Example 1. An Example of Estimate of the Situation Template

Specifically, the project should identify, analyze, and document the following process drivers in the EoS:

- Project mission, success criteria, history, and context.
- Project objectives.
- Inherited decisions and assumptions.
- Characteristics of the project's requirements and operations.

- Known constraints.
- Factors that affect the project's success.
- Comparison of possible courses of action.

All project team members should review and approve the EoS. Although the EoS does not have to circulate widely outside of the immediate project team, the senior manager should review, correct, and sign off on the EoS to ensure that the project's objectives are clear and correct (Charette 1989).

The project updates the EoS document during subsequent cycles. The project's current cycle objectives and potential alternatives are determined and included in the EoS document. The project's current cycle usually inherits a legacy from the previous cycles; specific objectives defined and alternatives selected in these early cycles might become problems to be solved in the later cycles. This legacy drives the current *cycle process*.

#### 3.2.1.1.6 Definition Review

A project uses a definition review meeting as a mechanism for validating and reaching consensus on the approach definition produced for the project and the specific cycles. In order to proceed, project team members must reach a firm agreement on the approach for the current stage of development. During the first cycle, the project examines, authenticates, and approves the definition review for the project level approach. In subsequent cycles, the purpose of the review changes to focus on the approach defined for the current cycle. During the definition review, a project should ensure that:

- Project-level objectives and success criteria are feasible.
- Cycle-level objectives support project-level objectives.
- Cycle-level success criteria are feasible.
- Alternatives proposed are consistent with objectives and constraints.
- Alternatives proposed adequately address objectives and constraints.
- All the stakeholders agree to pursue the project and/or current cycle.

Project and/or cycle information may change as a result of the definition review. A project should document the rationale behind these changes in the meeting minutes.

### 3.2.2 The Second Quadrant: Analyze and Avert Risk

A project performs risk analysis and aversion to help identify, address, and eliminate risk items before they become either threats to successful software operation or major sources of rework" (Boehm 1989). Risks are the potential undesired outcomes of alternatives, constraints, and other factors, and the effect of these outcomes on objectives and success criteria. Risk analysis and aversion is a defensive management approach that focuses on what can go wrong and then attempts to keep it from occurring.

During the second quadrant, a project concentrates on the factors, or risks, that may oppose project or cycle success. Most software development projects inherit or introduce some degree of risk. A project

specifically addresses these risks in the second quadrant of the ESP model by evaluating the alternatives defined in the first cycle in terms of the project constraints. The primary result of this evaluation is a quantified list of risks for the project and/or current cycle. Risks can be quantified by risk exposure, which is the product of the probability of failure (the chance of an unfavorable occurrence, or the risk "bet") and the cost of failure (what happens if the risk "bet" is lost).

Also during the second quadrant, a project should select one of the alternatives determined in the first quadrant, but only after its risks have been analyzed and averted to the extent possible. It is not necessary (or even possible) for a project to eliminate all risks during product development. The intention is to limit the risk exposure so that the success of the project is not in jeopardy. If a project finds itself in a zero-risk situation, then the ESP model can become equivalent to other process models such as the waterfall, evolutionary, or transform models, as mentioned in Boehm (1988).

Availability of the following resources may facilitate the second quadrant risk analysis and aversion activities:

- ***Experienced Risk Analyst.*** The ESP model depends on risk management to determine appropriate software development activities and their sequence. Although the project manager can function as the risk analyst, a project may find it desirable to use the services of someone properly trained in risk analysis and management techniques, if possible.
- ***Budget for Risk Activities.*** DoD contracts specifying DOD-STD-2167A require some form of risk management (Department of Defense 1988, paragraph 4.1.4). As a general rule of thumb, expect to allocate the following portion of a project's software development budget for risk management activities (Charette 1991):
  - One to three percent for low-risk projects.
  - Three to five percent for medium-risk projects.
  - Five to seven percent for high-risk projects.

The specific activities of the second quadrant are:

- ***Identify Risks.*** Determine and categorize risks for the project and/or current cycle.
- ***Analyze Risks.*** Determine the likelihood of occurrence and consequence of each risk, and rank the risks.
- ***Evaluate Risks.*** Identify and evaluate possible risk aversion strategies.
- ***Risk Review.*** Review the risk identification, analysis, and evaluation activities by project team.
- ***Plan Risk Aversion.*** Select and plan for the most appropriate risk aversion strategy.
- ***Commit to Aversion Plan.*** Commit formally to the selected risk aversion plan (all stakeholders).
- ***Avert Risks.*** Perform to risk aversion strategy.

The second quadrant risk analysis and aversion activities are graphically depicted in Figure 11, and are further defined in the subsections below.

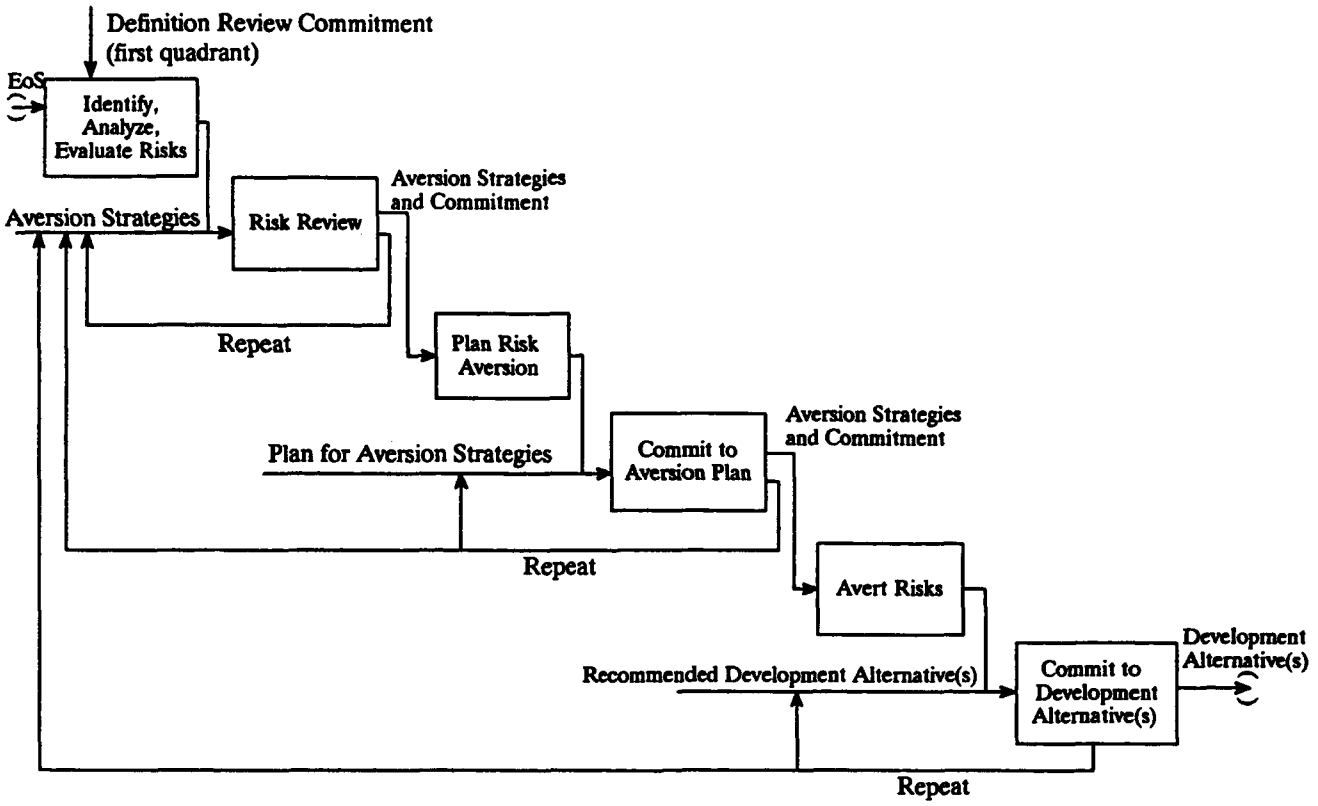



Figure 11. Risk Analysis and Aversion Activities

### 3.2.2.1 Identify Risks

A project generally performs risk identification by analyzing historical data or by gathering opinions (Ashley 1990). Some risk factors may be inherent in the objectives or constraints of the cycle, but others depend on the particular alternatives selected. Risk taxonomies that identify typical project risks are available in Boehm (1989, 117); U.S. Air Force (1988); Charette (1990, 216-59).

To identify a risk, its characteristics must be recognized. Three characteristics that are present in any risk (Charette 1989) are shown as the "three Cs" of the risk triangle illustrated in Figure 12.

- **Cost** is the unfavorable outcome if an uncertain event happens. That outcome must have a direct impact in order to be considered a risk (e.g., loss of profit, forfeiture of contract, layoffs, loss of prestige, or inability to gain a market). If there is no impact, gain, or loss, then the event is not a risk.

- There must be some reasonable **chance** of the uncertain event occurring in order for it to be considered a risk. If there is no chance of it occurring, then it is not a risk. If the chance of the event occurring is absolute, then the outcome of that event cannot be avoided and must be accepted as a constraint.

- A risk occurs only if there is a **choice** involved. Having no choice, or no options, means that an outcome cannot be avoided and must be accepted as a constraint.

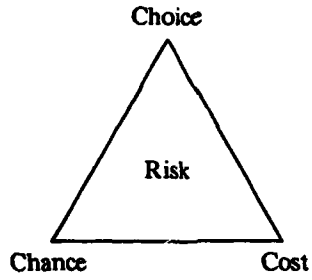



Figure 12. The Elements of Risk

Once identified, a project should categorize all relevant risks. Categorization helps determine and understand the underlying characterization and root causes of the risk. Possible mechanisms to categorize risk can be obtained from Charette (1989, 111) and U.S. Air Force (1988).

#### 3.2.2.2 Analyze Risks

During risk analysis, a project attempts to estimate the likelihood of occurrence and the exposure to potential loss (i.e., the consequence) of the identified risks. Specifically, a project attempts to determine the following for each identified risk:

- ***Probability of Occurrence.*** An estimation of the probability of an outcome occurring, based on project-specific measurement scales.
- ***Cost of Occurrence.*** An estimation of consequences based on the perceived cost to obtain resources and perform any actions that might be necessary should the risk occur. The cost may also reflect any gain, if applicable.

Charette (1989, 120-4) and the U.S. Air Force (1988) describe several approaches for developing measurement scales. Once risks are quantified, they can be compared, ranked, and communicated. In addition, a project can identify and analyze the relationships and impacts that risks may have on one another. A visual representation of risks can be a useful communication and documentation tool. One risk visualization method, the iso-risk contour, is explained in Charette (1989).

#### 3.2.2.3 Evaluate Risks

During risk evaluation, a project identifies risk aversion strategies and examines the impact of the risk aversion strategy on a specific risk. A risk aversion strategy attempts to reduce the cost and/or probability of risk occurrence to an acceptable level. Risk aversion strategies generally fall into one or more of the following categories:

- ***Risk Reduction.*** Sometimes called risk abatement or mitigation, this strategy attempts to reduce either the likelihood of a risk occurring and/or the magnitude of the risk. For example, a delivery date risk may be reduced by extending the schedule to provide more time.
- ***Risk Protection.*** This type of strategy attempts to lessen the impact of a risk, should it occur. Insurance is an example of risk protection. In the software development environment, a business area may make a practice of continually bidding contracts in order to protect against the risk of current contract cancellation.

- ***Risk Transfer.*** This approach attempts to reallocate risks to areas better able to handle them. For example, a project has a requirement that assumes existing knowledge of the client's *domain.* Several risks have been identified as a result of this requirement since no in-house expertise exists for the client domain. The risk aversion strategy in this case may be to subcontract the requirement to a consultant with domain knowledge who can handle the risk more easily.

- ***Risk Contingency Fund.*** This approach establishes a reserve of project resources that is set aside for use if any of the identified risks occurs. The most common reserved resources are time and money. For example, when generating the project schedule, extra time may be allocated to high-risk tasks that may require additional effort to solve.

- ***Risk Acceptance.*** There may be some cases where no cost-effective strategies are available to avert a risk, and the strategy is to do nothing intentionally and accept the consequences.

It is possible for risk aversion strategies to introduce new risks that will detract from the anticipated benefits. For example, extending a schedule to reduce a delivery date risk may introduce a risk of contract noncompliance. Therefore, the impact of the risk aversion strategies must also be identified, analyzed, and evaluated.

#### 3.2.2.4 Risk Review

The Risk Review activity is an opportunity for a project team to review and comment on the results of the risk identification, analysis, and evaluation activities. In many cases, the most effective use of time is for the project manager and/or risk analyst to identify, analyze, and evaluate risks independently and then submit the results of the activities to the rest of the team for review. Review comments may identify the need to repeat some or all of the previous risk activities.

#### 3.2.2.5 Plan Risk Aversion

A project should plan risk aversion strategies so that their results are available in time to plan for the development activities that depend on them. High-level cost and schedule for each risk aversion strategy should be estimated and evaluated. Ideally, the project team should be involved in this activity to help evaluate and select the best risk aversion strategy(ies).

#### 3.2.2.6 Commit to Aversion Plan

Commit to aversion plan is a mechanism for a project to formally brief stakeholders on the results of the previous risk activities and to solicit their commitment to the risk aversion strategies selected by the team. Identified risks should be discussed in terms of probability and cost of occurrence. The risk aversion strategies for these risks should be described in terms of:

- Is the strategy feasible?

- Does the strategy reduce risk to an acceptable level?

- Will the strategy negatively impact another risk?

- What is the potential impact of new risks, if any, introduced by the strategy?

- Does the strategy support cycle and/or project objectives and success criteria?
- Are the tactics and means for implementing the strategy consistent with cycle and/or project constraints?
- Is the strategy cost-effective, or will it waste project resources on unnecessary or non-beneficial risk aversion?

If a project team has already selected and committed to a specific risk aversion strategy, the project manager can present the rationale behind the selection; however, the risk aversion strategy committed to as a result of the risk review may not be the one initially recommended by a project team. If this is the case, risk activities may need to be repeated, as appropriate.

#### 3.2.2.7 Avert Risks

During risk aversion, a project performs the activities supporting the risk aversion strategy committed to in the previous step. Risk aversion activities may include prototyping, simulation, studies or surveys, comparative evaluations, evolutionary development, and other techniques as appropriate. Complicated or longer-term risk aversion activities may spin off into their own spirals and be managed independently from the main-line development discussed in the third quadrant. The results of the risk aversion activities should justify a particular *development alternative*, such as a specific method, strategy, or approach. Once a development alternative is selected, a project should plan and schedule technical activities in detail.

#### 3.2.2.8 Commit to Development Alternative

A project presents the results of the risk aversion activities to appropriate stakeholders and solicits commitment to the development alternative or approach selected. In some cases, the development alternative selected as a result of the risk aversion activities may not be committed to by senior management. If this is the case, risk activities may need to be repeated, as appropriate.

### 3.2.3 The Third Quadrant: Develop Product

A project plans, schedules, and performs the technical activities for the cycle in the third quadrant. The activities should be driven from the development alternatives committed to in the second quadrant. The results of performing the activities should directly support the cycle objectives and success criteria defined in the first quadrant.

The specific activities that a project performs in the third quadrant are:

- ***Plan and Schedule.*** Identify, organize, schedule, and assign resources to technical activities after any risks associated with development alternatives for the cycle have been averted.
- ***Commit to Plan.*** Review and commit to the cycle plans.
- ***Develop and Verify Product.*** Perform the activities for the cycle to produce artifacts or advance product maturity, and verify that the artifacts or advanced product maturity meet requirements, cycle objectives, and success criteria.

- ***Monitor and Review.*** Monitor and review the technical development activities as they are performed.
- ***Technical Product Review.*** Review product or part of product developed to ensure that cycle objectives and success criteria were met.

Each of the third quadrant product development activities are graphically depicted in Figure 13, and are further defined in the subsections below.

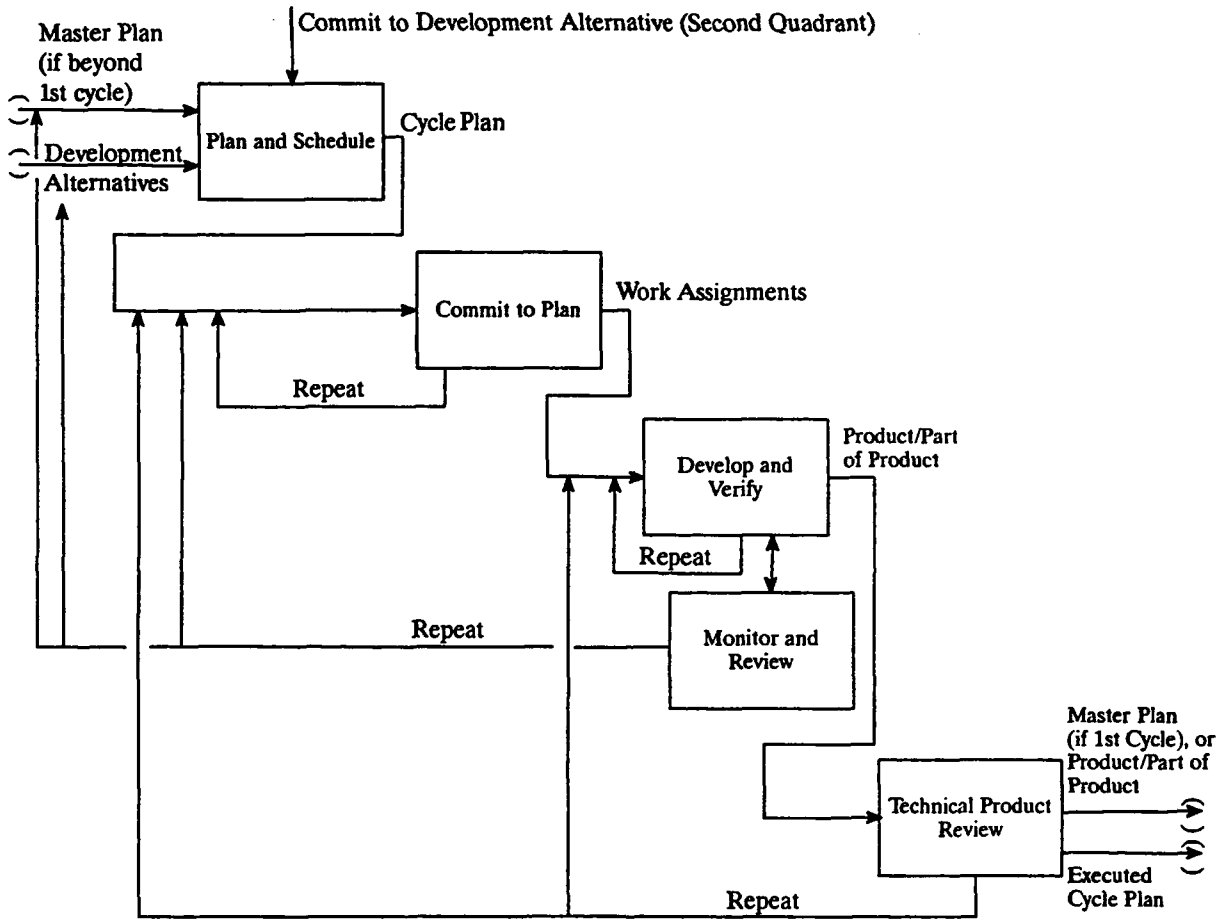



Figure 13. Develop Product Activities

### 3.2.3.1 Plan and Schedule

Although basic management functions of planning, monitoring, and controlling are performed continuously, a project identifies, organizes, and allocates resources for specific technical activities for the cycle once the cycle risks have been averted. The main output of the plan and schedule activity is a detailed cycle plan. The plan should:

- Select the activities to be performed in the current cycle.
- Define/redefine activities and supporting artifacts, if necessary.
- Select and allocate methods, practices, and tools.

- Estimate and allocate resources.
- Sequence the activities.
- Define *work packages*, or the lowest WBS level, for the key activities defined for the current cycle.

A project selects activities based on the cycle objectives, success criteria, constraints, and development alternatives. For example, a cycle objective may be to complete the preliminary requirements document, and the cycle success criteria may be to obtain document approval from the client. Activities selected for the cycle will most likely include activities for requirements identification, analysis, definition, and documentation; iterative client reviews; and final review and approval. In another case, the second quadrant commitment to a particular design method alternative constrains the selection of activities to those which support the selected design method.

A project can define activities and their supporting artifacts, if necessary, but a better approach might be to reuse, and adapt as appropriate, activity definitions from sources such as the organizational process definition, industry standards such as IEEE STD 1074 and DOD-STD-2167A, and various industry guidebooks and textbooks. In some cases, all or some of the activity and artifact definitions may be driven by a project constraint, such as client-required conformance to DOD-STD-2167A, or by specific activity definitions used by the development methods selected. A project should attempt to define activities in a consistent format, perhaps by using a specific notation. Activity and artifact definitions are documented as part of the engineering and project procedures, discussed in Section 3.2.4.3.

An activity is enactable when it is defined and sequenced relative to other activities, and the project selects and allocates the supporting methods, practices, tools, and resources. Specific methods, practices, and tools may be selected as part of a development alternative, or they may be driven by project or organizational objectives and constraints, such as a business area objective to standardize on a certain configuration management tool. Otherwise, a project should identify and allocate the development methods, practices, and tools when defining or adapting activities and artifacts since methods, practices, or tools may drive the selection and definition of activities. Descriptions of development methods, practices, and tools are documented as part of the engineering and project procedures, discussed in Section 3.2.4.3.

A project may estimate and allocate the resources to support activities based on project or cycle objectives, success criteria, constraints, or development alternatives. For example, computer resources may be limited to what is currently available in-house, a fixed-price contract may drive resource determination and allocation, or a specific verification method may require certain automated resources and/or team member participation.

The way a project sequences activities can be driven by several factors:

- Sequence can be driven by any ordering of activities required by a development alternative.
- Sequence is often partially determined by activity inputs and outputs. If a detailed design document is a necessary input to a coding activity, then the activity that produces the detailed design document should be performed before the coding activity, though not necessarily immediately before.

- Development alternatives that avert resource risks and constraints also drive activity sequencing. For example, since it may take several hours to construct a complete system from its source code, activities may be sequenced so that compiles can run overnight.

- Personnel allocation and leveling also drives the activity sequence by determining the order of activities that allocates team members at optimal levels throughout the cycle.

If a project needs to track costs to the work-package level, then work packages can be defined and opened as part of selecting, defining, resourcing, and sequencing activities.

#### 3.2.3.2 Commit to Plan

Commit to plan is an opportunity for a project team to review and comment on the results of the cycle planning and scheduling activities. The project manager may draft the cycle plan and submit the results of the activities to the rest of the team for review. The cycle plan may be updated in response to team comments.

Once a project team has committed to the detailed cycle plan, the project manager may want to brief senior management formally on the direction the current cycle development activities will take. This briefing may be a natural extension of any scheduled, periodic internal management reviews, or may need to be scheduled separately. As a result of committing to the cycle plan, work packages are opened and assigned.

#### 3.2.3.3 Develop and Verify Product

At this point in the third quadrant, a project develops the product or part of the product, analyzes the results for technical merit, and verifies results against cycle objectives and success criteria. During the first cycle of any spiral, a project always develops the Master Plan in the third quadrant. During subsequent cycles, a project will develop whatever product(s) and/or part of the product that is planned and scheduled for the current cycle.

A Master Plan brings order to the software development process and reduces wasted effort. Plans are generally used as guides; they are predictions of future events and outcomes that embody many assumptions. A Master Plan should attempt to predict the events that can keep the project from achieving its objectives and describe how best to address these events should they occur. Actual events should be monitored and compared to predictions as the project progresses, discrepancies should be analyzed, and the Master Plan should be updated accordingly.

The Master Plan is not a single document, but rather the complete set of planning documents that defines and guides the process of a software development project. At a minimum, the Master Plan produced during the first cycle of a spiral should include:

- Project-level EoS.

- Process model for the spiral.

- Project-level risk analysis and aversion plan.

- Budget allocation to the cost account level.

- Detailed plan for executing the first cycle ("plan the plan").
- High-level plan for the remainder of the spiral based on known objectives, success criteria, constraints, and development alternatives.

The Master Plan could also include:

- Software development plan.
- Hardware development/acquisition plan.
- Estimated cycle plans, or place holders for the cycle plans.
- Engineering and project standards and procedures.
- Development environment standards.
- Financial documents (WBS, etc.).
- Other required planning documents, such as a test plan.

The Master Plan is updated or expanded upon during subsequent cycles. For example, cycle-specific objectives, alternatives, constraints, risks, and alternatives identified by the activities of the first, second, and third quadrants of each cycle will result in an update to, or further elaboration of, the EoS, the risk analysis and aversion plan, and the estimated cycle plans. Other updates to the Master Plan take place in the fourth quadrant, as described in Section 3.2.4.4.

During the cycles that follow the first cycle, a project will execute main-line, or traditional, software development activities in the third quadrant, such as requirements, design, code, integration, and test. A project performs the specific activities in the third quadrant that are identified as a result of executing the first two quadrants and the planning and scheduling activity of the first part of the third quadrant. Differences may or may not be obvious to a developer who is used to a more sequential approach. Ideally, development under the ESP model would evolve to a more conventional, waterfall-like development model if risk is negligible or is reduced to an acceptable level. In general, this type of process evolves if major project risks are addressed by the time software architecture is completed.

Because three of the four quadrants of the ESP model execute project-management-related activities, there may be an implication that a project spends more time managing than developing. This is a misconception. Once risk is reduced, a project performs activities of the first, second, and fourth quadrants relatively quickly during each cycle and spends the majority of time and resources performing the third-quadrant product in development activities such as design, code, integration, and test.

A project should perform verification as an integral part of the third quadrant development activities. Verification ensures that the artifacts or advanced product maturity produced by the development activities meet quality requirements, as well as cycle objectives and success criteria. Verification techniques may be informal walkthroughs or reviews, or formal inspections.

#### 3.2.3.4 Monitor and Review

Monitoring and review maintains management control over the software development process. A project tracks, reports, and manages costs, schedules, and risks. A project also monitors and reviews

cycle performance, which is closely tied to the measurements generated by each development activity performed in the third quadrant.

A project can use several methods or tools to support the monitoring activity. Traditional scheduling tools help track progress and how actual costs and schedules compare to estimates. Risks can be monitored using a *risk referent*. A risk referent is a measure of acceptable risk for each individual and overall project risk. As development activities are performed, the risks associated with the activities are monitored and tracked against that referent.

Project reviews facilitate a project team interchange that affects the direction of the project. Periodic project reviews demonstrate to the project team and other stakeholders that the project is under control. The project also uses them to obtain a commitment from stakeholders to proceed with the plan. At these reviews, a project makes decisions to reallocate resources, replan schedules, or reassess risks for the current cycle activities.

#### 3.2.3.5 Technical Product Review

The technical product review is a project team review of the product or part of the product developed to ensure that cycle objectives and success criteria were met. This review is an opportunity for team review; however, the ongoing Monitor and Review activity should have identified and resolved any factor that might have had a negative impact on meeting cycle objectives and/or success criteria.

When a project evaluates the product technically at the end of each development step (quadrant 3), the probability of successful delivery of the final product should increase. Failure to reduce overall project risk after a few attempts should be cause for concern. The notion that overall project risk should decrease is reflected in the ESP model as an increase in probability of success as you move from the third to the fourth quadrant for each cycle. Note that risk does not disappear entirely, so a project should constantly identify and mitigate risks. Only when the product is successfully installed, and a project has been released from liability, does the probability of success become a certainty.

### 3.2.4 The Fourth Quadrant: Manage and Plan

During the fourth and last quadrant, a project takes stock of progress based on the outcome and lessons learned during the cycle, compares actual results against the cycle objectives, re-evaluates and updates master planning documents, and decides what to do next. The planning and management activities validate the work performed in the cycle and adjust the project strategy based on the information that is newly available. The measurable objectives, or the success criteria defined during the first quadrant and monitored during the third quadrant, are critical to this set of activities. They are used to judge whether the cycle is complete or needs to be iterated, or if an alternative should be modified or abandoned.

The specific activities of the fourth quadrant are:

- ***Baseline Product.*** Place the product(s) or part of the product produced as a result of executing the third quadrant development activities under configuration control.

- ***Review Progress.*** Evaluate cycle plan actuals versus estimates, success criteria, and lessons learned. Update process drivers, including project objectives, success criteria, alternatives, constraints, risks, estimates, and other information.

- ***Develop/Update Engineering and Project Procedures.*** Update engineering and project procedures, if necessary, based on lessons learned during the cycle.
- ***Update Master Plan.*** Update all planning documents, as necessary, to record actual progress, reflect lessons learned, update estimates based on actual data, and update process drivers.
- ***Management Review and Commit to Proceed.*** Review updates to the Master Plan and commit to proceed with next cycle.

Each of the fourth quadrant planning and managing activities are graphically depicted in Figure 14, and are further defined in the subsections below.

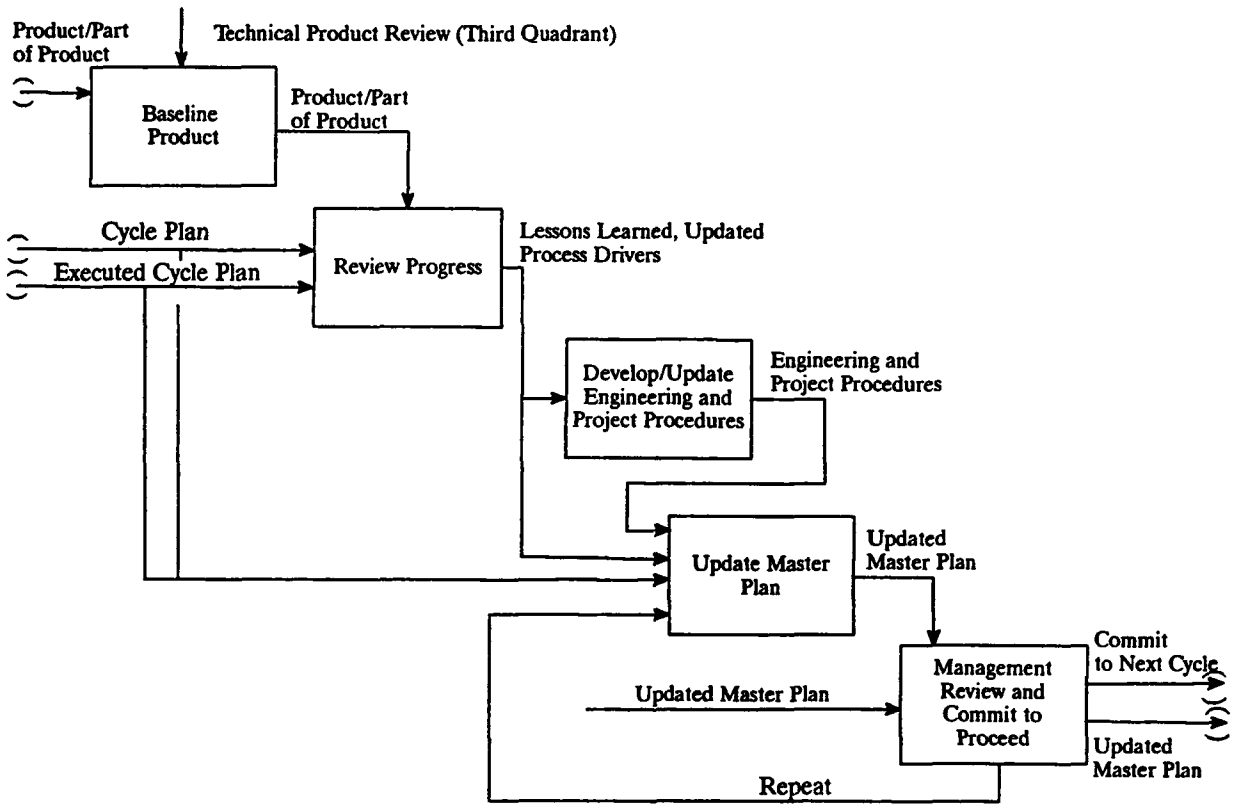



Figure 14. Manage and Plan Activities

### 3.2.4.1 Baseline Product

Once cycle objectives and success criteria are met and the product or part of the product developed in the third quadrant is approved by the team during the technical product review, then a project baselines the product(s) or part of the product produced during the cycle. The baseline should be placed under configuration control using an appropriate method and/or tool.

### 3.2.4.2 Review Progress

During the review progress activity, a project evaluates cycle plan actuals versus estimates, success criteria, and lessons learned. As a result of the evaluation, a project updates process drivers such as project objectives, success criteria, alternatives, constraints, risks, estimates, and other information.

#### 3.2.4.3 Develop/Update Engineering and Project Procedures

Engineering and project procedures include activity and artifact specifications and method descriptions. They should be adapted from an organizational process definition if one exists. Procedures may be driven by a client requirement to follow a certain standard or by the specific development alternative(s) selected in the second quadrant.

During the first cycle, a project can define and/or adapt whatever candidate engineering and project procedures are appropriate to the current level of understanding about the project and any development alternatives committed to. The project can define remaining procedures as appropriate when the project progresses through subsequent cycles and commits to development alternatives. For example, a project may know during the first cycle that it will use a particular configuration management (CM) method and tool, and can define or adapt the CM procedures accordingly. The design procedure may not be defined or adapted until the second cycle, however, after the project evaluates the candidate design methods and commits to an alternative.

Once defined, a project can update the engineering and project procedures, as necessary, based on lessons learned, updates to process drivers, or a change in development alternative. For example, a cycle may have followed a code inspection procedure that required that code be formally inspected before being compiled. The experience of the cycle may have shown that some errors could be caught more quickly and consistently by a machine than by a human, so the procedure was updated to remove the compiling restriction for future coding cycles.

#### 3.2.4.4 Update Master Plan

The last activity a project performs before committing to proceed with the next cycle is to update all planning documents, as necessary, to reflect lessons learned, estimates based on actual data, and updated process drivers. The Master Plan becomes a living document, as well as serving as the repository of project process history and changes.

#### 3.2.4.5 Management Review and Commit to Proceed

The commit concept, at the end of the fourth quadrant, is one of the most important in using the spiral model. It is similar in purpose to a baseline. All stakeholders in a project should be briefed on the results of the current cycle and on any changes in plans. They should agree with the decisions made regarding what to do next. Stakeholders include project management at all levels, as well as the project team. The review may also include customer representatives. The purpose of the review is for a project to determine whether project-level objectives, alternatives, and constraints are still appropriate and on-track, agree on the objectives and success criteria for the next cycle, and commit resources to that cycle.

## 3.3 SUBSPIRALS

A project generally starts with a single, main spiral, but the development effort can be partitioned into multiple, parallel spirals to address subproducts or specific risk areas. The subspiral is generally a smaller version of the main spiral, but is otherwise identical. Within each spiral the cycles may be sequential or parallel, but there is no implied ordering among the cycles of different spirals. A project often initiates subspirals in the second quadrant to address a specific risk. For example, using Ada

as the programming language may be one of the development alternatives under consideration for a specific project; however, a performance problem associated with its use is suspected. The risk aversion strategy to address this potential risk is the initiation of a parallel subspiral. The Ada risk-reduction subspiral will define its own cycles as it addresses its objective (to determine if Ada can satisfy the performance requirements). The information produced by the Ada risk-reduction spiral will feed the main spiral, most likely affecting design decisions in the software architecture.

In addition, the processes for parallel development of subproducts, such as multiple CSCIs, can be defined and managed through the use of subspirals. Parallel development of subproducts is initiated and planned for in the fourth quadrant and is controlled by the same Master Plan, i.e., the Master Plan is developed for the whole system as a result of the first cycle of the main spiral. In the Master Plan, the decision to develop subproducts in parallel is made, documented, and committed to. During the subspirals, each cycle in the subspiral can produce an independent cycle plan; however, there is only one Master Plan for the system. The Master Plan is updated during the fourth quadrant of each cycle for each subspiral. In the case of the parallel development of subproducts, the main spiral may be used solely for subproduct management and integration activities, to develop the most important subproduct, or to develop the subproduct with the highest risk.

## 3.4 SUMMARY

The project's process, and the product that the process produces, is refined as each cycle of the ESP model is executed. Early cycles are generally small and usually concentrate on reducing risk and product planning; later spirals concentrate on product development. The very first cycle of the spiral focuses on plans for the project as a whole, while subsequent cycles concentrate on detailed planning based on cycle objectives, success criteria, and development alternatives. Some of the distinguishing features of the ESP model are:

- ***Explicit Determination of Process Drivers.*** Using the ESP model on any given project requires development and documentation of process drivers, including objectives, success criteria, alternatives, constraints, risks, and other information. Focusing on these drivers will help engineer a process that is appropriate for a project or cycle, i.e., this process identifies and defines the right development activities and assigns the best available methods and resources to support the activities in order to meet cycle, and ultimately project, objectives and success criteria.

- ***Risk Analysis.*** The ESP model incorporates explicit steps to analyze and manage risks that threaten the success of the project. The ESP model helps identify risk issues early in the project life cycle when resources are available to deal with them effectively. The risk-influenced sequence of activities contrasts with linear process models, which follow a strict sequence of activities that allows risks to be deferred or ignored until it is too late to act.

- ***Risk Aversion.*** The ESP model recognizes that there are several risk mitigation strategies and helps identify the best strategy for averting a risk. Risk aversion can attempt to reduce a risk by executing activities such as prototyping, modeling, simulation, benchmarks, and comparative evaluation, or by reallocating cost or schedule. In some cases, the risk aversion strategy may be simply to accept the risk and its potential consequences.

- ***Planning and Schedule Flexibility.*** The ESP model allows for a dynamic development process. Each cycle is planned and scheduled in detail as it occurs, and may define its own completion

criteria. In addition, the ESP model allows for management of concurrent subtasks that may progress somewhat independently of the principal tasks.

- ***Subspirals.*** The ESP model allows development of multiple, parallel spirals to address subproducts or specific risk areas, allowing immediate attention to critical or risky areas without slowing down progress on other parts of the project. In addition, subspirals allow parallel development of subproducts, such as multiple CSCIs.

- ***Metrics and Measures.*** The ESP model is metric and measurement driven. No cycle or spiral can complete unless measurable success criteria are defined, monitored, and met. Data such as actual and estimated cycle and project costs, schedules, risks, and performance are used to plan and execute the activities more effectively for the remainder of the project.

- ***Procedures and Standards.*** The ESP model helps to define a process compatible with required engineering and project procedures of any client and/or business area. In addition, the ESP model supports the latest Department of Defense acquisition policy. For example, DOD Instruction 5000.2 (Department of Defense 1991) advocates evolutionary development of command and control systems.

- ***Commitment to Progress.*** The ESP model requires explicit commitments to proceed with the project after each project cycle. This activity provides the following benefits:

  - It serves as a mechanism for controlling requirements changes.

  - It serves as a mechanism to identify previous commitments that a change impacts.

- ***Commitment by Stakeholders.*** The ESP model requires periodic review and approval of decisions affecting the product and process alternatives and risks. This may require frequent reviews, demonstrations, and walkthroughs. The customer and project staff may become involved in decision making more than is usual in traditional approaches.

- ***Mature Process.*** The ESP model provides an ideal framework for implementing the key practices of a mature process (SEI CMM).

Section 4 of this report will combine the ESP model concepts and principles discussed in this section with the process introduction material in Section 2 to describe evolutionary process engineering at the project level.

# 4. EVOLUTIONARY PROCESS ENGINEERING

Rather than having a single monolithic process that all projects must use, [organizations] will find that different projects will have differing needs.

Peter H. Feiler and Watts S. Humphrey,
*Software Process Development and Enactment: Concepts and Definitions*

## 4.1 OVERVIEW

An integral feature of the ESP model is the ability to engineer the best possible process for addressing project needs. When following the ESP model, a project engineers its process dynamically by continuously documenting, instantiating, enacting, and evolving its software development process definition in an evolutionary fashion as shown in Figure 15. Although dynamic, the ESP model specifically provides for orderly and controlled evolution of a baselined project process.

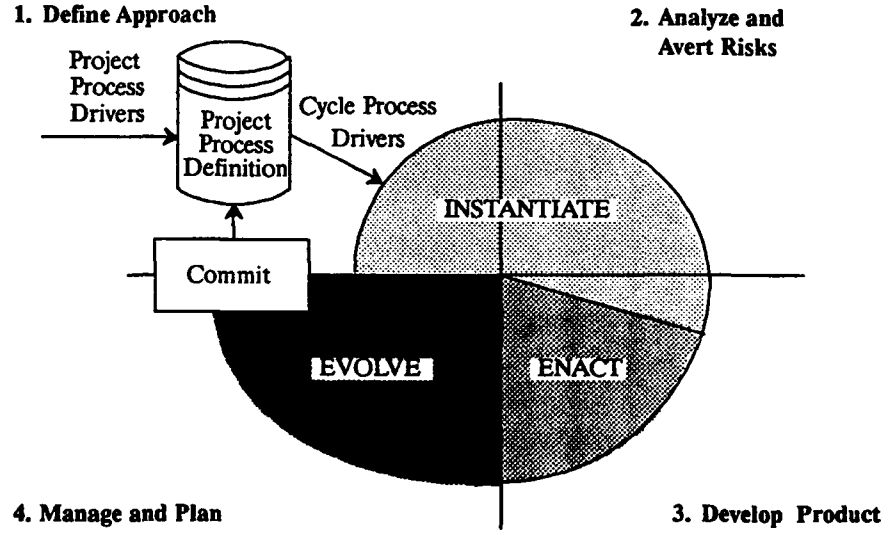



Figure 15. Evolutionary Project-Level Process Engineering

Evolutionary process engineering emphasizes the identification and continuous evaluation of key project and cycle characteristics with the potential for driving the process in some significant way. In the first cycle, a project's process definition is documented either by defining an overall process from scratch or by tailoring an organizational process definition to the level of detail appropriate to the current understanding about the project and its unique process drivers. In subsequent cycles, a project instantiates its project-level process definition by defining it to an enactable level of detail based on *cycle process drivers*, enacts or performs the cycle process, and evolves the process definition for the remainder of the project, based on information gained, lessons learned, progress to date, and early strategies and mitigation decisions.

A project can follow the ESP model to engineer its software development process regardless of the level of organizational process maturity, as measured by the CMM. Although doing so is time and resource intensive, a project can develop its project process definition without having an organizational process definition to use as a foundation. The project-specific process model, activity specifications, and method descriptions engineered as a result of following the ESP model can be abstracted and used by organizations at an ad hoc or repeatable process maturity level as input to developing an organizational process definition.

Experience shows that developing comprehensive software process definitions can be very expensive and time consuming, and it may be more desirable to develop general-purpose process definitions, together with techniques for reusing, tailoring, and enhancing them (Feiler and Humphrey 1992, 4). Projects that are a part of an organization at the defined level of the CMM or above are likely to spend less time and effort engineering its process because of the availability of:

- A standard organizational process definition.
- Tools, methods, and techniques, and supporting technology transfer mechanisms.
- An advanced and/or automated organizational software engineering environment.
- Historical technical and management data.

A project at a higher level of process maturity will generally tailor and instantiate the available organizational process definition and historical data into a project process definition as required by unique process drivers.

The sections below discuss how evolutionary process engineering is an integral part of the ESP model. The ESP model encompasses process engineering actions at the project level by:

- Documenting and baselining a project process definition in the first cycle.
- Instantiating the process for a cycle based on cycle process drivers.
- Evolving a project process definition as a project progresses and the process is enacted.

As organizations reach the highest levels of process maturity, it may be possible to optimize traditional process engineering activities by developing specific tailoring and instantiation guidance.

## 4.2 DOCUMENTING THE PROJECT PROCESS IN THE FIRST CYCLE

A project documents its high-level process by enacting the activities in the first cycle of the project's spiral to:

- Identify and evaluate project-level process drivers.
- Define the project process from scratch or tailor the organizational process definition, as required to address the drivers.

A project attempts to identify and evaluate those key process drivers that have the potential to impact the entire project. It then documents the process at a level of detail that reflects the current

understanding about the project and its identified process drivers. A project process definition is documented as a result of defining the overall process from scratch or of tailoring an existing organizational process definition, depending on the process maturity level of the organization.

### 4.2.1 Identify and Evaluate Project-Level Process Drivers

Process drivers can be viewed as the source of process definition or tailoring requirements that a project process must meet. Process drivers tell a project what the key considerations are when attempting to define, or tailor the organizational process definition to, a software development process that will best meet the project's needs. In the first quadrant of the first cycle, a project can attempt to identify project process drivers by analyzing available project information sources, such as:

- The project contract.
- The project Statement of Work.
- Customer data, such as:
  - Target environment.
  - Requirements.
  - Policies, procedures, standards, and regulations.
- User data, if different from customer data.
- Internal infrastructure data, if available, such as:
  - Organizational process definition.
  - Historical engineering data.
  - Current software engineering environment.

A project can also identify process drivers as a result of informal interviews with team members, experienced professionals in the organization, the customer, or appropriate consultants. A project organizes and documents the identified process drivers through the mechanism of the EoS. Specifically, the project documents project-level objectives, success criteria, alternatives, constraints, risks, and other considerations. See Section 3.2.1 for a more detailed description of the first quadrant activities that result in the creation or update of the EoS. See also Appendix A for a partial listing of potential process drivers.

When identifying and documenting objectives and success criteria in the EoS, a project can use the *Software Measurement Guidebook* (Software Productivity Consortium 1992b) for guidance on how to establish and monitor quantifiable objectives. By adding values to the quantifiable objectives, a project can derive the related success criteria.

During the second quadrant, a project identifies additional project and/or cycle process drivers in the form of risks. Some of these risks will be apparent from the information documented in the EoS; other risks can be identified as a result of using a risk taxonomy or by analyzing historical data and experience. Risks that have a high overall risk level, based on high probability and cost of occurrence, tend to be the process drivers which place the most significant requirements on the process. See Section 3.2.2 for more detail on the second quadrant risk identification activity.

Accurate and timely identification of key process drivers can determine project success or failure. Example 2 is a case of a project failing because it did not recognize one of its major process drivers in time.

The WIPEOUT program mission was initiated by the Army to take advantage of the rapid influx of real-time target data to coordinate and control air attacks on ground targets. The WIPEOUT contract was awarded to ABC, Inc., a large company with extensive real-time systems experience. The WIPEOUT project team at ABC, Inc. won the contract based on its solution to automate the creation of mission assignments: that is, the creation of detailed assignments for Air Force pilots would be created automatically in response to data gathered through troop reports, satellites, reconnaissance, ground surveillance, and other information-collection mechanisms. The automated mission assignments would provide the pilot with the minimum information necessary to conduct his mission, such as mission start time, flight coordinates, and exact time to release missiles.

The WIPEOUT project team worked closely with the Army, carefully analyzed requirements, and spent significant time creating, analyzing, and validating its preliminary design. The project team identified and considered many potential problems that could occur as a result of integrating several data sources and working with rapidly changing, and perhaps contradictory, data. The project produced a preliminary design document that the team felt addressed several of the potential problems and risks.

After two and a half years, and $15 million, the WIPEOUT project team was ready to brief the preliminary systems design to the program committee. Shortly after a thorough and professional briefing to the program committee, the project was cancelled.

**WHY?**

Although the WIPEOUT project team considered all the technical factors of the system, they failed to consider the organizational and political factors that affect their users. As a result, they did not interview any Air Force representatives or consider the potential reaction of pilots to the idea of having mission assignments automated. The WIPEOUT preliminary design briefing to the program committee was the first time that the Air Force saw the Army's plans for their pilots, and there was an uproar. The Air Force wanted their pilots thoroughly briefed on their assignments, including information on what the target was, exactly where the pilot was going, and who was going to be waiting for the pilot when he got there. The Air Force categorically refused to fly any missions created by the Army's WIPEOUT system.

Example 2. An Example of the Importance of Identifying Process Drivers

A project can expand the second quadrant activities, described in Section 3.2.2, to evaluate strategies for all identified process drivers instead of just those that are risks and to commit to a specific process alternative as a result. Strategies for addressing project process drivers may include:

- Developing a process model that identifies and orders the project's essential set of software development activities.
- Defining or tailoring activity descriptions and/or accomplishments in a project-specific way.
- Organizing or reorganizing activity sequences and relationships in a project-specific way.

- Determining high-level resource and/or responsibility allocations.
- Identifying potential methods, tools, and/or techniques.

A project evaluates how well each strategy addresses the relevant process driver, how the strategy might affect other process drivers, and the costs associated with each strategy. Example 3 describes how a candidate strategy for one key project process driver was significantly affected by another key process driver.

> Impact Corporation is a large company supported by both commercial and government contracts. One of the divisions, the Decisions Support Division (DSD), has recently won a multi-year, military contract for developing predictive cost models (PCM). A year prior to winning the new contract, DSD conducted an internal self-assessment as developed by the Software Engineering Institute (SEI) of Carnegie-Mellon University. The DSD process maturity was assessed at a level two. DSD management is committed to improving the process to a defined level of maturity, or level three of the CMM, as soon as possible. In the year since the initial assessment, the process improvement group of the DSD has drafted a divisional process definition in the form of a divisional guidebook. In order to validate the divisional process definition, all new projects are required to follow the new divisional guidebook.
>
> The PCM project identified its project process drivers and documented them in the EoS. One of the process drivers identified when analyzing the PCM contract was the external constraint to comply with DOD-STD-2167A, the military standard for defense system development. Several members of the PCM project are very familiar with DOD-STD-2167A since it has been a required standard on several of the government contracts won by Impact Corporation. The initial strategy proposed for addressing this process driver was to tailor the activities and artifacts required in the standard to the PCM project and use the result as the project process definition.
>
> When the PCM project team evaluated the proposed strategy, they realized that the strategy conflicted with another process driver: the internal constraint to use the divisional process definition. In addition, the potential strategy to tailor the divisional process definition to be compliant with DOD-STD-2167A as well as to address the other project process drivers was more costly in time and effort than the PCM project could feasibly absorb.
>
> Since there was no choice about complying with the external constraint, the project team had to compromise the internal constraint process driver. As a result, a strategy was evaluated to define the project process based on a combination of the tailored standard and the tailored divisional process definition. Whenever the PCM determined a conflict, the project had the authority to define its process in accordance with DOD-STD-2167A over the divisional process definition. The PCM project requested and received DSD management commitment to this strategy. In order to avoid this problem in the future, the DSD process improvement group was tasked to develop guidance on how to tailor the divisional process definition to comply with DOD-STD-2167A.

Example 3. An Example of Evaluating and Committing to a Strategy

A project should also carefully consider the technology transfer impact when evaluating strategies that involve new technology to the project, such as a new method, tool, or technique. A project should evaluate and commit to the technology transfer activities and supporting resources needed to introduce

the new technology into the project. By not incorporating orderly and defined technology transfer activities as an integral part of the project process definition, a project runs the risk of unsuccessfully addressing the process driver because the selected method, tool, or technique alternative is neither understood nor used effectively by the project team.

A project can use the *Using New Technologies: A Software Engineering Technology Transfer Guidebook* (Software Productivity Consortium 1992c) when identifying and evaluating strategies that involve new technology. The guidebook provides detailed guidance on identifying and selecting new technologies, as well as defines the process activities necessary to transfer the technology for maximum use and benefit. Example 4 describes a project that did not transfer effectively the technology it had committed to as a result of a process driver.

When the Phoenix project defined its high-level project process definition, it identified a significant risk of changing requirements. The project team determined that this risk had a high likelihood of occurring and the potential for affecting seriously the success of the entire project. In order to help address this risk, the team decided to use a formal requirements engineering method.

The Phoenix project was a part of a level two organization. Although the project team had access to an organizational guidebook, the organization had not yet supported the organizational process definition with candidate tools, methods, and techniques, and supporting technology transfer mechanisms. As a result, the Phoenix project spent time and resources to identify, evaluate, and procure industry requirements methods, and to commit to a specific requirements method alternative.

The project tailored the organizational level process definition in order to make the activities of the requirements method alternative an integral part of the project-level process definition. However, the Phoenix project did not include technology transfer activities and supporting resources in the project-level process definition, such as:

- Develop and commit to a technology transfer implementation plan.
- Implement user support mechanisms such as training, consulting, and scheduling adjustments for the learning curve.

As the project progressed, the Phoenix project team attempted to enact the defined requirements engineering activities, but they were unsuccessful without adequate support for installing, learning, and using the technology. Because the team had schedule and deliverable constraints that had to be met, they abandoned the attempt to use the formal requirements engineering method and quickly put together a requirements document in an ad hoc fashion.

As suspected at the beginning of the project, requirements changed and significantly impacted the cost, schedule, and quality of the product. The project team agreed that much of the negative impact might have been avoided if the requirements method had been used effectively.

Example 4. An Example of a Process Driver Strategy Involving New Technology

At the end of the second quadrant, project-level process drivers are identified, strategies evaluated, and development alternatives committed to, if any. A project documents its project process by defining one from the ground up, or by tailoring an existing organizational process definition in order to meet the requirements imposed by the project-level process drivers and corresponding strategies.

### 4.2.2 Define/Tailor the Project-Level Process Definition

As a result of enacting the third and fourth quadrant activities of the first cycle of the spiral (described in Sections 3.2.3 and 3.2.4), a project produces its Master Plan, or the complete set of high-level planning, oversight, and support documents that define and guide the project's software development process. The project develops its Master Plan in accordance with the requirements imposed by the project-level process drivers and strategies. A project-level process definition can be documented through a subset of Master Plan documents, as shown in Table 1.

Table 1. Subset of Master Plan Documenting Project-Level Process Definition

| This Master Plan Document (from Section 3.2.3.3): | Can Provide this Part of the Project Process Definition: |
|---|---|
| Project-level EoS. | Process drivers and requirements for defining or tailoring the project-level process definition. |
| Spiral process model. | The set of key project activities. See Figure 9 for an example spiral for a project. |
| Project-level risk analysis and aversion plan (expanded to include process drivers other than risk). | Process definition and/or tailoring requirements, strategies for addressing the requirements, and process alternatives. |
| Software development plan. | Project-level WBS and supporting size, schedule, staffing estimates, relationships between WBS activities, and personnel, practices, and tools, if known. |
| Budget allocation to the cost account level of the WBS. | Cost estimates for each major activity. |
| Estimated cycle plans, or place holders for the cycle plans. | Scheduling and staffing estimates broken down to cycle activities and tasks; relationships between cycle activities and tasks within each activity; and supporting personnel, practices, and tools. |
| Engineering and project standards and procedures. | Description of how to perform each activity and the method, practice or tool selected to support the activity; when to start and stop each activity; and what each activity accomplishes or produces. |

Process drivers and resulting strategies can drive the documentation of a project-level process definition regardless of organizational process maturity level. A project at a low process maturity level organization, which must document its process from scratch, can view process drivers and resulting strategies as process definition requirements. A project at a higher process maturity level organization can view process drivers and strategies as requirements for tailoring an organizational process definition to a project-specific one. Example 5 shows how two projects, at organizations of different process maturity levels, defined their project's process.

At the end of the first cycle, a project process definition should be documented to the level of detail appropriate to the current understanding of the remainder of the spiral, committed to by the project stakeholders, and baselined. During subsequent cycles, a project instantiates, or creates an enactable process of, the process definition for the current cycle based on process drivers identified for the cycle.

Turtle Corp., a level one organization, had a project with an objective of design for reuse. The project developed its process definition based primarily on the activities, methods, and techniques intended to support designing for reuse. It spent time, money, and resources to research and develop a design for reuse method, specify the method activities, train the entire staff, and invest in a reuse library tool. Because of the amount of resources spent to address this one process driver, the project did not have the resources to define and support many other process activities, such as testing and CM. As a result, the design activities went smoothly, but integration, testing, and other basic life-cycle activities were performed in an ad hoc manner. Because the software was well-designed, fewer fatal errors were introduced than expected; however, the last several months of the project progressed in a chaotic fashion.

Hyper International, a level three organization, had a project with the same reuse objectives. Hyper International recently published the third version of its internal guidebook, which documents the organizational process definition. The project tailored the design and reuse library procedures included in the guidebook to address specifically its design for reuse objective. At the same time, the project tailored other parts of the guidebook to the level of detail appropriate to the known process drivers and evaluated the methods, tools, and training available in the organization. As the project progressed, it committed to resources, methods, and tools, and evolved its process based on lessons learned, impact of previous strategies and decisions, and newly identified or more thoroughly understood process drivers. This project's process addressed all life-cycle development, management, and support activities in a relatively constant manner. However, the project team felt that the time spent tailoring the organizational process definition was excessive and should somehow be reduced.

Example 5. An Example of Organizational Process Maturity and Project-Level Process Definitions

## 4.3 INSTANTIATING THE PROJECT PROCESS FOR A CYCLE

A project instantiates the project-level process definition produced during the first cycle for the current cycle by:

- Identifying and evaluating cycle process drivers.
- Documenting the current cycle process to an enactable level of detail.

A project attempts to identify and evaluate those cycle process drivers that have the potential to impact the current cycle. Then it documents the cycle process to a level of detail that includes all of the elements needed for enactment.

### 4.3.1 Identify and Evaluate Cycle Process Drivers

Cycle process drivers can be viewed as the source of instantiation requirements that a project process must meet. Current cycle process drivers are usually a decomposition of the project process drivers that either specifically affect the current cycle or are inherited from the previous cycles. The exception to this is the first cycle, where the main objective of the cycle process is to produce the Master Plan. During the first quadrant of the current cycle, a project uses the plans, metrics, and status reports produced or updated during the previous cycles to identify cycle process drivers. As a result, the project updates the EoS produced in the first cycle with the current cycle objectives, success criteria, alternatives, constraints, risks, and other considerations.

During the second quadrant, a project identifies additional cycle process drivers in the form of risks; analyzes strategies; and commits to a specific risk aversion plan as described in Section 3.2.2.6. Cycle process alternatives should be defined in terms of:

- The necessary set of development, planning, and support activities.
- How to perform each activity.
- What each activity will accomplish or produce.
- Who will perform the activity.
- How much time and effort the project thinks the activity will take.
- What methods, practices, and tools will be used to the perform the activity.

The project team should agree that the selected process alternative represents the best solution for meeting cycle objectives and success criteria, averting cycle risks, and addressing other cycle process drivers.

A project tends to have the most flexibility for addressing key process drivers in the first cycles. The decisions and commitments made, and the actions taken, as the project progresses create a legacy that may place restrictions on process alternatives in later cycles. The greater the legacy, the less process flexibility a project may have. In some cases, however, the best strategy may be to wait until later in the process before evaluating candidate strategies and/or committing to a specific process alternative. For example, a project may identify one or more potential design methods, but may not want to commit to a specific method until the impact of requirements is more clearly understood.

### 4.3.2 Document the Cycle Process

During the plan and schedule activity in the first part of the third quadrant, a project documents the cycle process to an enactable level of detail by producing a cycle plan, described in Section 3.2.3.1. The cycle plan should describe the process alternatives selected to address the cycle process drivers and should be consistent with the framework provided in the project-level process definition. If a project cannot produce a cycle plan that is generally consistent with the project-level process definition, then the project should consider the inconsistency as a risk, evaluate any potential impact on subsequent cycles, and commit to evolving the project-level process definition as appropriate.

A project can consider the process for the current cycle instantiated when:

- Cycle process drivers, including cycle objectives and success criteria, are identified and approved by the project team.
- Process alternatives to address the cycle process drivers are selected, defined, and committed to by the project team.
- The cycle plan is documented and details the:
  - Size, cost, schedule, and errors estimated for the product or part of product to be developed during the cycle.
  - Size, cost, schedule, and error data collection mechanisms.
  - Risk monitoring mechanisms.

- Selected methods, practices, tools, and other process alternatives for supporting the cycle.
- Cycle activities required to meet cycle process drivers and alternatives, and how the activities relate to each other.
- Project and engineering procedures that describe how to perform the cycle activities, and which are defined or tailored as required to meet cycle objectives and success criteria, accommodate cycle alternatives, satisfy cycle constraints, and avert cycle risks.
- Allocation of resources and responsibilities to support all cycle activities.
- Work packages for the key cycle activities.

- The project stakeholders have approved the cycle plan and individual work assignments.
- The cycle plan is baselined and distributed to all stakeholders.
- The project manager opens cycle work packages and authorizes charges to them.

A simple tool that a project can use to represent an instantiated process graphically is a PERT chart. Figure 16 shows how a PERT chart can represent the process in terms of the set of cycle activities and supporting activity sequence, estimated durations, start and end dates, and resources. Some PERT chart tools capture more information than is shown in the figure, such as fields to capture the WBS code and cost data; free text fields for brief descriptions of activities, artifacts, and/or supporting methods; and historical data and metrics.

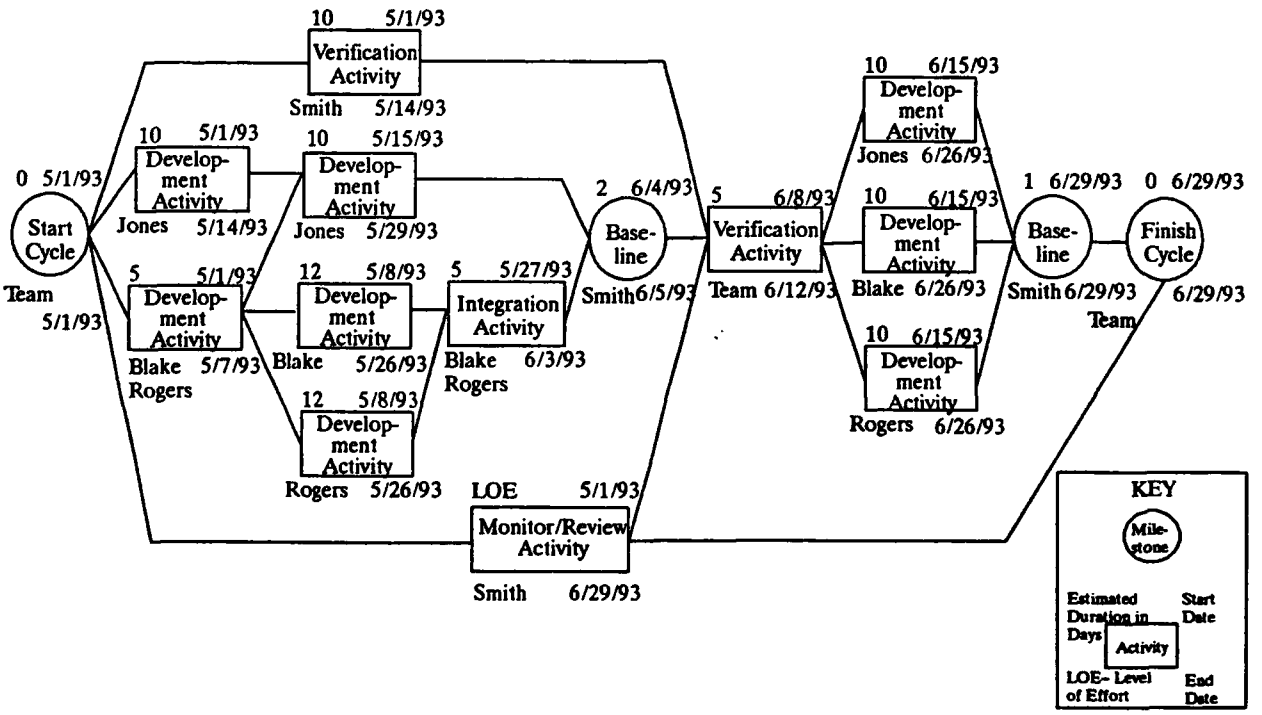



Figure 16. Basic Process Representation With a Project Evaluation and Review Technique Chart

## 4.4 ENACTING THE CYCLE PROCESS

A project enacts the baselined cycle process in the third quadrant of the current cycle by:

- Performing to the baselined cycle plan to develop and verify the product or part of the product.
- Monitoring and reviewing the cycle activities as they are being performed.
- Analyzing the results for technical merit.
- Verifying results against cycle objectives and success criteria.

In the first cycle, the product developed as a result of enacting the process is generally the Master Plan. In subsequent cycles, the enacted process should produce the product or part of the product necessary to meet the cycle objectives and success criteria. See Section 3.2.3 for detail on third quadrant activities.

At a minimum, a project should collect the following basic project data as an integral part of enacting the cycle process:

- Actual total labor effort for each planned and scheduled cycle activity.
- Actual number and type of product components (e.g., modules, documentation) produced.
- Actual size of each product component.
- Actual number, severity, and source of errors found during verification activities.
- Actual risk increase or reduction of each high-priority risk and of the overall project risk.
- Product change data.
- Lessons learned.

The enacted cycle process, together with lessons learned and the data collected by monitoring and reviewing the cycle activities, are used to review and subsequently evolve the process for the remainder of the project. A project can use the *Software Measurement Guidebook* (Software Productivity Consortium 1992b) for specific guidance on defining, estimating, and collecting size, cost, schedule, and error data.

## 4.5 EVOLVING THE PROJECT PROCESS

Feiler and Humphrey (1992, 5) state that as projects evolve and members of project teams gain experience with the process, there will be many ideas for improvement; therefore, it is important to establish mechanisms to obtain continual user feedback to guide process repair and evolution. The ESP model provides for these mechanisms in the fourth quadrant by specifically including process activities intended to help a project review the process to date and to plan and manage process evolution. It is important for a project to control process review and evolution carefully since the results of these activities often place instantiation requirements on subsequent cycles or may even require rework of previous activities.

Process review and subsequent evolution can be performed quickly within the current cycle if evolving the process impacts the **current** cycle only: that is, as a project enacts the current cycle process in the third quadrant, it may discover the need to reallocate resources, replan schedules, or reassess risks in order to meet cycle objectives and success criteria. If the impact of addressing the new needs affects only the current cycle activities, then a project can quickly review the process drivers, evolve the cycle process accordingly, and baseline the evolved cycle process. Example 6 describes how a project was able to review and evolve the cycle process definition quickly to address a new process driver discovered as a result of enacting the cycle process.

Impact Corporation is a large company supported by millions of dollars in government contracts. One of the divisions, the Decisions Support Division (DSD), has an ongoing, multi-year project for developing predictive cost models (PCM). The PCM project is currently enacting preliminary requirements activities. A planned work in progress review with the customer uncovers a risk; the customer cannot visualize how the PCM project is going to meet some of the key reporting requirements. As a result, the customer is not very comfortable with the work in progress.

The PCM project holds a team meeting to discuss the risk to its cycle objective of passing preliminary design review (PDR). The team decides that developing a prototype of the user interface will help the customer to understand better and to visualize how the team planned to meet reporting requirements. Jane, one of the team members, is proficient in an in-house, PC-based prototyping tool, and estimates that it would take her three weeks to prototype screens and report formats, inspect the prototype with team members, and incorporate changes. Jane is currently scheduled to draft part of the preliminary requirements document over the next four weeks.

Tom, one of Jane's teammates, volunteers to assume responsibility for Jane's part of the preliminary requirements document since there is a great deal of integration involved in their assignments. Tom estimates that the added responsibility will consume the slack time he currently had on the schedule, but not add any additional time, providing that Jane is able to review work in progress. The project manager updates the cycle plans and PERT chart to reflect the new tasks and assignments. Tom's progress is carefully monitored since he is placed on the schedule's critical path as a result of using available slack time to meet new responsibilities.

Although the cycle process was not enacted as originally defined, the PCM project met its cycle objective and passed PDR.

Example 6. An Example of Process Evolution Within a Cycle

After meeting cycle objectives and success criteria, and baselining the product or part of product produced during the current cycle, the project reviews progress to date, including:

- The basic project data collected when enacting the cycle process.
- How well the enacted process accomplished cycle objectives and success criteria.
- How closely the project followed the baselined process as instantiated for the cycle.
- Identified process strengths, weaknesses, and defects.
- Newly discovered or better-understood objectives, alternatives, constraints, risks, or other process drivers.

As a result of the review, a project can evolve its project process definition to correct defects or weaknesses, or to meet new process drivers, by:

- Updating the EoS to document new or better-understood process drivers.
- Updating engineering and project procedures to reflect lessons learned, updated process drivers, or change in development alternative.
- Updating project process model and schedule, staff, and budget estimates as necessary to reflect changes based on actual cycle data.
- Getting approval of the evolved version of the project process definition by stakeholders.
- Baselining the evolved version of the project process definition.

Once enacted, a review of the process to date may place significant evolution requirements on the project's process for the remainder of the project. For example, a project may:

- Uncover or clarify process drivers that impact subsequent cycles, such as actual labor hours remaining to support the project.
- Uncover a process driver that requires rework of already enacted activities, such as a failed quality metric, a misunderstood requirement, or a fatal design flaw.
- Evolve the project process definition in a way that impacts the remainder of the project, such as updating a policy or procedure.

Example 7 shows how the project-level process definition and a process driver discovered during a previous cycle's process review placed a legacy of instantiation requirements on the current cycle.

Once a project has successfully met all the project-level objectives and success criteria, then it can take advantage of the fourth quadrant process activities to review the final project process definition and enactment data as a whole to identify trends, statistics, and improvements to use as input to the next project or to evolve the organizational process definition.

## 4.6 OPTIMIZING EVOLUTIONARY PROCESS ENGINEERING

Engineering a process that meets a project's needs is a critical part of producing high-quality software within budget and on schedule; it can also be time consuming and costly. Although level three of the CMM requires guidance on why, when, and how to tailor the organizational process definition, the reality is that identifying project process drivers and then manually tailoring and instantiating the project process can be a resource- and time-intensive task. In addition, tailoring guidance and resulting procedures often require that the purpose and rationale for any tailoring of the existing organizational process definition be documented, justified, and approved.

Specific tailoring and instantiation guidance based on fundamental project process drivers might be a solution for minimizing the time and resources projects must spend to define their project-unique process adequately. Tailoring guidance based on inputs such as the estimated project size, cost, schedule, risks, and available resources and assets can help a project define its process in the first cycle by providing answers to questions such as:

The TRI-277 project had a 12 month duration constraint. During the first cycle, the project team evaluated different verification methods. Although most of the project team was trained in the formal inspection method, the schedule did not allow for formal inspections of requirements, design, and code for all components. The project team decided to perform informal peer reviews instead of formal inspections. The project team documented its project-level process definition and peer review procedures during the third and fourth quadrants of the first cycle.

The project progressed by evolving and enacting the process definition for each cycle. During the cycle where the team enacted integration activities, almost three times as many errors were discovered in module A as in any other module. This discovery introduced a process driver for the current cycle: a risk of failing a current cycle objective for passing integration tests.

In order to determine and correct the causes of the errors, the team decided on a risk mitigation strategy to inspect formally the requirements, design, and code documents for the module. They discovered that module A attempted to address the correct requirements, but was much larger and more complex than the rest of the modules. As a result, the team instantiated the current cycle process definition to include a subspiral for redesigning and recoding module A. Resources were obtained and reallocated, and formal inspection activities were made a mandatory part of the subspiral process for module A. The activities for the rest of the modules were evolved and enacted during the current cycle in accordance with the high-level process definition.

The TRI-277 project team met its current cycle objective for passing integration tests as a result of its concentrated effort to redevelop module A. The team agreed, however, that they could have done a better job of estimating module size and complexity in order to identify high-risk modules earlier. The team also agreed that, at a minimum, high-risk modules should be formally inspected.

Example 7. An Example of a Process Driver Legacy

- What is the right set of process activities, and the appropriate process model, for the project?
- How should the activity specifications be tailored?
- What should the artifacts be?
- How should resources be allocated to activities?
- What methods, tools, and techniques should be used, and how should their use be customized?

As the project learns more about its process drivers in subsequent cycles, further guidance is needed to assist with process evolution activities. Guidance may be required on how to evolve the project process definition, given the current level of understanding and knowledge about the process drivers, such as actual product measurements, including cycle costs, schedule, risks, resources, and assets.

Providing tailoring and instantiation guidance based on specific variations and priorities of process drivers would appear to be a monumental task. First, the standard organizational process definition and supporting environment should encompass all basic life-cycle, management, and support activities. Second, much knowledge and experience should be accumulated as a result of using the organizational process definition in order to determine the best response(s) to each process driver. Third, it is likely to take significant time and effort to define standard process drivers and to derive appropriate

conditions, criteria, and solutions to instantiate mechanically the organizational process definition in order to produce a project-specific definition.

These difficulties are currently being examined by the Software Technology for Adaptable Reliable Systems Center (STARS) and the SEI. Specifically, the STARS/SEI team has proposed to define and document process assets, or the components for constructing project-specific processes. The initial definition and documentation effort will include the collection, cataloging, analysis, partitioning, distillation, and synthesis of software processes submitted by industry, government, and academic organizations. The resulting process assets including adaptation, tailoring, installation, and evolution guidance will be piloted and tested. Once tested, the process assets and corresponding guidelines will be made available to the industry (Over 1991, 45-60).

The Synthesis model, described in Section 2.2.1.4, could also be used to address the problem of process tailoring and instantiation. In order to maximize the benefit of the organizational process definition, the Synthesis model advocates defining process at the domain level. Domains can cut across the industry to the extent that different companies in the industry have similar business objectives, produce similar products, or serve a similar customer base. For example, each of several large companies may have domains that build similar aerospace systems or subsystems, such as aircraft avionics, missile controls, or helicopter flight simulators.

By developing domain process definitions, it may be possible to identify the specific process drivers, such as the objectives, alternatives, constraints, and risks, often associated with that particular domain. Domain experience and knowledge can be used to determine the candidate solutions available to address those drivers and determine the criteria for tailoring the domain process definition based on predefined process drivers. Well-defined variations of activities, methods, practices, tools, and resources can be predetermined to be the optimum strategy for specific process drivers or combination(s) of process drivers.

As a result of working through a set of predetermined questions about process drivers, the domain process definition can be tailored and methods, practices, tools, and resources appropriately selected and allocated in order to produce an enactable process for a specific application project within the domain. Because application processes will generally vary from the domain process in well-defined ways, it may be possible to automate the activity of tailoring and instantiating the domain process definition in order to produce a process that is application specific. In addition, no tailoring justification would be necessary, unless an application wished to tailor the domain process in a way different from the predetermined and approved variations.

## 4.7 SUMMARY

Using the ESP model results in evolutionary process engineering at the project level by:

- ***Defining a Project's Process in the First Cycle.*** A project identifies and analyzes its project-level process drivers in order to define the project-level process definition or tailor the organizational process definition, as required to address the drivers. A project physically documents the project-level process definition in the first cycle through a subset of the Master Plan documents. The project-level process definition is defined to the level of detail appropriate to the current understanding of the known process drivers and the remainder of the spiral activities, approved by stakeholders, and baselined.

- ***Instantiating the Project Process for a Cycle.*** A project instantiates the process definition for the current cycle as required to address cycle process drivers. Cycle process drivers include cycle objectives, success criteria, constraints, alternatives, and risks. Cycle process drivers also incorporate the legacy inherited from previous cycles. A cycle plan documents all the information required to enact the cycle process. If the cycle plan is not generally consistent with the project-level process definition, then the project should consider the inconsistency as a risk, evaluate any potential impact on subsequent cycles and commit to evolving the project-level process definition as appropriate. Once defined and approved by stakeholders, the cycle process is baselined.

- ***Enacting the Cycle Process.*** A project enacts the baselined cycle process in the third quadrant of the current cycle by performing to the cycle plan, monitoring and reviewing the cycle activities as they are being performed, analyzing the results for technical merit, and verifying results against cycle objectives and success criteria.

- ***Evolving the Project Process.*** A project reviews previous cycle data and evolves the process definition for the remainder of the spiral based on information gained, lessons learned, progress to date, and early strategies and mitigation decisions. Process review and evolution often place instantiation requirements on subsequent cycles. In some cases, a project can quickly evolve and instantiate the process definition for the current cycle if the impact of the evolution only affects that cycle. In all cases, the evolved cycle and/or project process definition is approved by stakeholders and baselined.

A project can follow the ESP model to engineer its project-level process regardless of the level of organizational process maturity. Projects at higher process maturity level organizations may spend less time producing the project process definition, however, than low process maturity organizations since:

- Tailoring an organizational process definition should take less time than documenting a new one.

- Instantiating a project-level process definition should be easier if the organization collects and supports process data, methods, tools, training, and consulting.

- Time and effort spent evolving a project-level process definition in order to resolve process problems and errors should decrease as the organizational process definition evolves through use and improvement.

As organizations begin to optimize their organizational process definitions, they can begin to develop and automate mechanical tailoring and instantiation guidance based on specific variations and priorities of process drivers.

# APPENDIX A. PROCESS DRIVERS

> Many of the problems on software projects arise from the mismatches between the process...used by the project and the project's real-world process drivers.
>
> Barry Boehm and Frank Belz, *Experiences with the Spiral Model as a Process Model Generator*

## A.1 OVERVIEW

This appendix provides reference information on process drivers. As discussed in Section 2.3.1, process drivers determine the desired characteristics of the development process. After identifying process drivers and the resulting desired process characteristics, you define, tailor, and instantiate the process definition to contain the characteristics. In the five tables below (Tables 2 to 6) process drivers are grouped into five different classes, depending on the source of the process driver. Taken together, the tables provide a fairly comprehensive list of software process drivers; however, no one list can cover all drivers for all situations.

Use the five tables to assist with defining, tailoring, and instantiating a software development process definition at the organization level for a particular business area or at the project level for a specific project. For each process driver that applies in the left column, note the corresponding desired process characteristic in the right column. As you look through each one of the tables, compile a list of desired process characteristics for your particular business area or project. Choose a process model (see Section 2.2.1) from your company's process archives or from another source that best fits your list of desired process characteristics. For the characteristics that indicate local tailoring, you tailor and instantiate your process definition to accommodate your unique process drivers.

In order to achieve gains through reusable work products and activities, strive to standardize your process as much as possible at the organizational level, leaving only the flexibility to tailor and instantiate as required to meet variances at the project level. The more parts of the process that are standardized and used repeatedly, the more effective and more efficient the organization will become at using and improving these parts.

## A.2 PRODUCT REQUIREMENT PROCESS DRIVERS

Table 2 identifies several process drivers relating to product requirements from the user. Each individual product requirement can potentially impact the architecture of the software development process.

Table 2. Product Requirement Process Drivers

| Process Driver | Desired Process Characteristic |
|---|---|
| ***Requirements understanding.*** Low understanding of requirements. | Prototyping.<br>Evolutionary development.<br>Planning for change. |
| ***Requirements stability.*** Changing requirements. | Incremental development.<br>Evolutionary development.<br>Planning for change. |
| ***Operational life cycle.*** System will be fielded for a long time. | Evolutionary development.<br>Planning for change. |
| ***Performance requirements.*** Difficult requirements for processing speed or memory or input/output usage. | Prototyping.<br>Modeling.<br>Benchmarking. |
| ***Maintenance/support requirements.*** Difficult requirements for maintenance or support. | Prototyping.<br>Enhanced support documentation (local tailoring). |
| ***System interfaces.*** Numerous or difficult system interfaces. | Prototyping (local tailoring). |
| ***Early capability.*** Early capability needed. | Incremental development.<br>Prototyping.<br>Evolutionary development. |
| ***Phasing.*** Required phasing with system increments. | Incremental development |
| ***User interface.*** Difficult or unstable user interface requirements or compatibility with existing user interface. | Prototyping. |

## A.3 ARCHITECTURE AND TECHNOLOGY PROCESS DRIVERS

Table 3 identifies several process drivers related to the architecture or underlying technology of the delivered product.

Table 3. Architecture and Technology Process Drivers

| Process Driver | Desired Process Characteristic |
|---|---|
| ***Size.*** Large or very large application. | Incremental development. |
| ***Growth envelope.*** Potential for large growth in size and/or diversity of system. | Late implementation binding.<br>Planning for change. |

Table 3, continued

| Process Driver | Desired Process Characteristic |
|---|---|
| ***Architecture understanding.*** Low understanding of system architecture. | Prototyping.<br>Accommodates early system analysis.<br>Late implementation binding.<br>Planning for change. |
| ***System nucleus.*** High-risk system nucleus. | Incremental development.<br>Planning for change. |
| ***Underlying technology.*** Available technology such as COTS or fourth generation language cover the system's growth envelope. | Accommodates underlying technology (local tailoring). |

## A.4 DEVELOPMENT CAPABILITY PROCESS DRIVERS

Table 4 identifies several process drivers related to the availability of development resources.

Table 4. Development Capability Process Drivers

| Process Driver | Desired Process Characteristic |
|---|---|
| ***Staff/facilities.*** Existing staff capabilities or existing development facilities. | Works with existing staff or facilities (local tailoring). |
| ***Hardware/software.*** Available development hardware or software. | Capability to use existing hardware or software (local tailoring). |
| ***Methods.*** Existing development methods. | Accommodating existing methods (local tailoring). |
| ***Concurrency.*** Need to work simultaneously in different phases. | Concurrency capability. |

## A.5 ORGANIZATIONAL ENVIRONMENT PROCESS DRIVERS

Table 5 identifies several process drivers related to the corporate organization surrounding the software development process.

Table 5. Organizational Environment Process Drivers

| Process Driver | Desired Process Characteristic |
|---|---|
| ***Organizational structure.*** Existing organizational structure. | Works within existing organizational structure (local tailoring). |
| ***Subcontractor relationships.*** Need to accommodate various subcontractor relationships. | Includes mechanisms for monitoring and controlling subcontractors (local tailoring). |
| ***Non-written agreements.*** Need to work without written agreements. | Includes mechanisms for approval/coordination that do not require paperwork (local tailoring). |
| ***Descriptive power.*** Need for visibility into steps or activities of the process. | Well-defined and well-documented process model. |

Table 5, continued

| Process Driver | Desired Process Characteristic |
|---|---|
| ***Visibility to process management.*** Need visibility to quality, configuration management, etc. | Well-defined process model that includes process management activities. |
| ***Control.*** Need for control over a lengthy development effort. | Disciplined process and well-defined process model with adequate process controls. |
| ***Continuous process improvement.*** Enables continuous process improvement. | Measurable.<br><br>Able to change and to incorporate lessons learned. |
| ***Domain experience.*** Lack of domain experience. | Accommodates for staff's lack of domain experience (local tailoring). |
| ***Standards or policies.*** Need to comply with organization standards or policies. | Complies with standards or policies (local tailoring). |
| ***Legal issues.*** Need to comply with company strategy for avoiding legal issues. | Works within legal constraints (local tailoring). |

## A.6 CUSTOMER ENVIRONMENT PROCESS DRIVERS

Table 6 identifies several process drivers related to the customer's interface with the software development process.

Table 6. Customer Environment Process Drivers

| Process Driver | *Desired Process Characteristic* |
|---|---|
| ***Fixed cost/schedule.*** Hard cost or schedule constraints. | Design-to-cost or design-to-schedule capability. |
| ***Dynamic cost/schedule.*** Unstable or changing cost or schedule constraints. | Flexibility to accommodate cost or schedule changes. |
| ***Milestones.*** Loose or fuzzy customer milestones. | Prototyping.<br><br>Incremental development. |
| ***Requirements.*** Changing or ambiguous customer requirements. | Prototyping.<br><br>Incremental development.<br><br>Planning for change. |
| ***Certifiability.*** Need to demonstrate specific process activities were performed. | Well-defined process model with mechanisms to verify completion of activities. |
| ***Contractual constraints.*** Difficult contractual constraints. | Works within constraints (local tailoring). |
| ***Standards or policies.*** Need to comply with customer standards or policies. | Complies with standards or policies (local tailoring). |
| ***Specified hardware or software.*** Customer-specified hardware or software. | Works with specified hardware or software (local tailoring). |
| ***Change control.*** Customer configuration management or engineering change proposal procedures. | Complies with customer procedures (local tailoring). |

# APPENDIX B. EVOLUTIONARY SPIRAL PROCESS ACTIVITY SPECIFICATIONS

Never tell people how to do things. Tell them what to do and they will surprise you with their ingenuity.

General George S. Patton

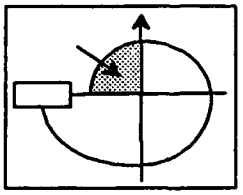
## B.1 OVERVIEW

This section defines the key activities essential to generating a process using the ESP model. Table 7 gives a complete list of the ESP activities specified in this section.

Table 7. Evolutionary Spiral Process Activities

| Quadrant | Activity Name |
|---|---|
| 1. Define Approach | Define Project/Cycle |
| | Develop/Update Estimate of the Situation |
| | Definition Review |
| 2. Analyze and Avert Risks | Identify Risks |
| | Analyze Risks |
| | Evaluate Risks |
| | Risk Review |
| | Plan Risk Aversion |
| | Commit to Aversion Plan |
| | Avert Risks |
| | Commit to Development Alternative |
| 3. Develop Product | Plan and Schedule |
| | Commit to Plan |
| | Develop and Verify Product |
| | Monitor and Review |
| | Technical Product Review |
| 4. Manage and Plan | Baseline Product |
| | Review Progress |
| | Develop/Update Engineering and Project Procedures |
| | Update Master Plan |
| | Management Review and Commitment to Pursue Project |

The format for the activity definitions uses an enhanced set of ETVX attributes. See Section 2.2.2.1 for a description of the ETVX notation. A description of the activity format is given below.



This section maps the activity back to the ESP model by identifying the quadrant, and the point in the quadrant, where the activity is begun.

***Performers*** This section identifies who will execute the activity.

***Inputs*** This section identifies what artifacts will be used during the performance of the activity. It is not necessary for all the inputs to exist in order to begin the activity. It is possible for only a subset of inputs to exist during the first iteration through the activity; as more inputs become available, subsequent passes are made through the activity.

***Outputs*** This section identifies what artifacts will be generated as a direct result of performing the activity.

***Entrance Criteria*** This section identifies prerequisites that the input artifacts must satisfy in order to begin the activity.

***Exit Criteria*** This section states what conditions the output artifacts must meet before the activity is considered complete. It is assumed that each activity satisfies the following exit criteria:

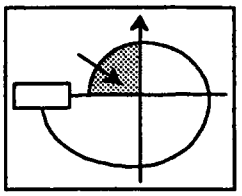
- Each artifact generated by the activity has been baselined, verified, and approved.
- Each artifact generated by the activity conforms to the organizational policies, standards, and procedures.
- Each artifact generated by the activity has collected process, risk, and quality information.

***Description*** This section describes the tasks to be performed during the execution of the activity.

***Measurables*** This section describes the measurables to be collected during the execution of the activity. The measurables are used to monitor activity progress, to aid in estimating the effort required to complete the activity, and to improve the process.

Note that every activity measures total activity labor effort: that is, actual calender start and end dates, team members who worked the activity, and total labor hours expended on the activity.

## B.2 DEFINE PROJECT/CYCLE



This activity is begun in quadrant one, Define Approach.

***Performers***

This activity is performed by the technical lead, project manager, risk analyst, and contract manager.

***Inputs***

The inputs to this activity in the **first cycle** are:

- Contract.
- Statement of Work.
- Documents describing the customer:
  - Target environment.
  - Policies, procedures, standards, and regulations.
  - Requirements.
- Internal infrastructure data, including:
  - Internal culture and investments.
  - Organizational process definition, if available.
  - Historical engineering data, if available.
  - Current software engineering environment and support.

The inputs to this activity in **subsequent cycles** are:

- Updated and approved Master Plan documents.
- Project metrics and status reports.
- Configuration data.

***Outputs***

The output of this activity in the **first cycle** is an informally documented project-level and first-cycle approach definition.

The output of this activity in **subsequent cycles** is an informally documented cycle-level approach definition.

***Entrance Criteria***

The entrance criteria for this activity during the **first cycle** are:

- Signed contract or authorization to proceed with the project.
- Resources are allocated to enact the first cycle activities.

The entrance criteria for this activity during **subsequent cycles** are:

- All stakeholders have approved updated Master Plan documents.
- Management gives authorization to proceed in current cycle.

***Exit Criteria***

The exit criteria for this activity in the **first cycle** are that project-level and first cycle process drivers are defined in terms of objectives, success criteria, alternatives, constraints, risks, and other project considerations, to the level of detail appropriate to the current level of understanding about the project.

The exit criteria for this activity in **subsequent cycles** are:

- The project has analyzed and incorporated the legacy inherited from previous cycles into the cycle definition.
- The project has defined the cycle-level process drivers in terms of objectives, success criteria, alternatives, constraints, risks, and other project considerations.

***Description***

The project approach definition in the **first cycle** establishes the ground rules for measuring progress at the project level. Identify, collect, and analyze input artifacts to define the project in terms of its project-level process drivers. If desirable, interview the project stakeholders for additional insights and information. Define the following process drivers with enough precision to establish objective measures for determining if the project is meeting expectations or if the expectations are infeasible:

- ***Objectives.*** What is to be accomplished during the project.
- ***Success criteria.*** How to know when the project is completed.
- ***Alternatives.*** Different ways to meet the objectives.
- ***Constraints.*** Limitations on alternatives.

At a minimum, try to identify those key process drivers with the potential to impact the entire project.

The cycle approach definition in both the **first cycle** and **subsequent cycles** establishes the ground rules for measuring progress at a detailed level for the current cycle. Team members can also interview the project stakeholders for additional insights and information. Define the cycle in terms of the current cycle process drivers. In addition, define the following with enough precision to establish objective measures for determining if the cycle is meeting expectations or if the expectations are infeasible:

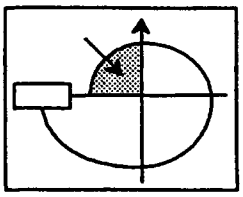
- ***Objectives.*** What is to be accomplished during the current cycle.
- ***Success criteria.*** How to know when the project current cycle is completed.
- ***Alternatives.*** Different ways to meet the objectives.
- ***Constraints.*** Limitations on alternatives.

Analyze the updated Master Plan documents to determine the current cycle process drivers.

***Measurables***

The measurement to be taken for this activity is the total activity labor effort.

## B.3 DEVELOP/UPDATE ESTIMATE OF THE SITUATION



This activity is begun in quadrant one, Define Approach.

***Performers***

This activity is performed by the technical lead, project manager, and risk analyst.

***Inputs***

The input to this activity in the **first cycle** is the project definition and first cycle definition.

The input to this activity in **subsequent cycles** is the cycle definition.

***Outputs***

The output of this activity in the **first cycle** is the draft Estimate of the Situation (EoS).

The output of this activity in **subsequent cycles** is an EoS updated to include current cycle process drivers and other information.

***Entrance Criteria***

The entrance criterion for this activity in the **first cycle** is that the project has identified and analyzed project-level and first-cycle process drivers.

The entrance criterion for this activity in **subsequent cycles** is that the project has identified and analyzed cycle process drivers.

***Exit Criteria***

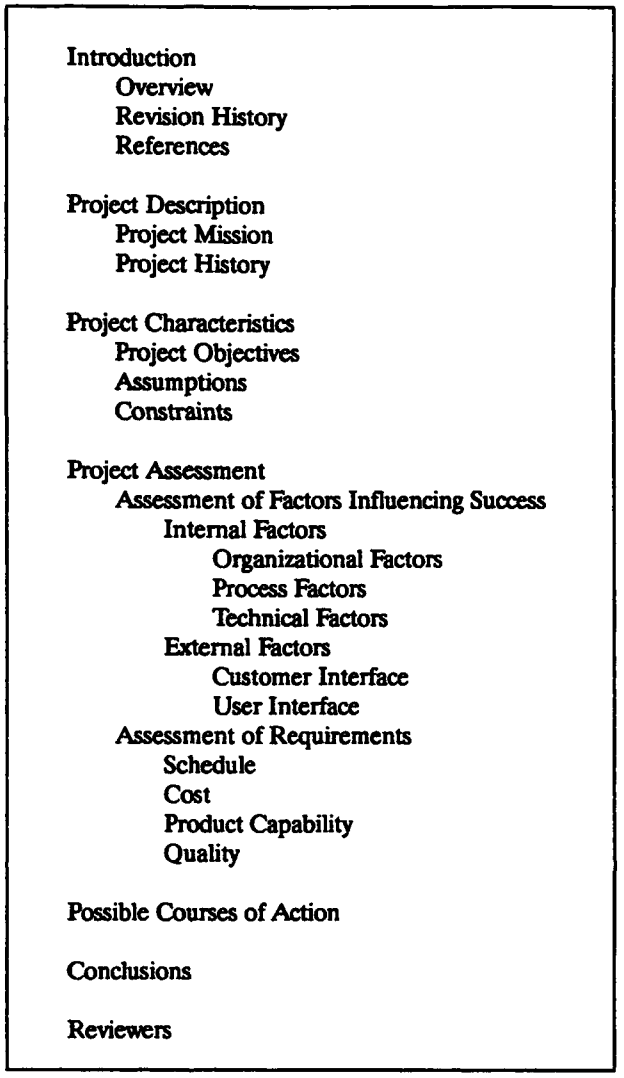
The exit criterion for this activity in the **first cycle** is that the EoS is updated to the level of detail appropriate to the current level of understanding of the project and its process drivers.

The exit criterion for this activity in **subsequent cycles** is that the EoS is updated to the level of detail appropriate to the current understanding of the cycle and the cycle process drivers.

***Description***

During the **first cycle**, the Develop/Update EoS activity is conducted primarily to set the context for planning and running the project. It describes why the organization is undertaking this project, what risks and assumptions are inherited, and what factors are critical to success. Consider political, economic, and technical issues in developing the EoS.

Organize and document the information collected in the Define Project/Cycle activity as shown in Example 8, an EoS template adapted from Charette (1989).



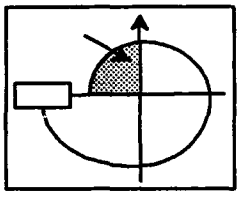

Example 8. An Example of Estimate of the Situation Template

Although the EoS need not be circulated widely, seek out the senior corporate manager responsible for the project. Have that person review, correct, and sign off on the EoS to ensure that the project's goals are clear and correct.

In the first cycle and **subsequent cycles**, update the EoS with the current cycle objectives, success criteria, constraints, potential alternatives, risks, and other considerations. Be sure to analyze the legacy from the previous cycles and document any impact to the current cycle in the updated Master Plan documents.

***Measurables*** The measurement to be taken for this activity is the total activity labor effort.

## B.4 DEFINITION REVIEW



This activity is begun in quadrant one, Define Approach.

***Performers*** This activity is performed by all project stakeholders, including technical lead, engineer, contract manager, quality manager, user, project manager, risk analyst, and customer.

***Inputs*** The input to this activity is the draft or updated Estimate of the Situation (EoS).

***Outputs*** The outputs of this activity are:

- Meeting minutes.
- Approved EoS.

***Entrance Criteria*** The entrance criteria for this activity are:

- The EoS document is drafted for the project or updated for the cycle.
- All individuals with an interest in the success of the project or their representatives are participating.

***Exit Criteria*** The exit criterion for this activity is consensus to proceed with the project.

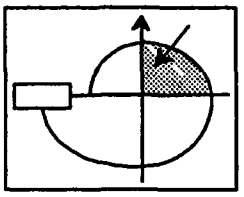
***Description*** To proceed, the project obtains firm agreement on what to pursue for this stage of development. The project reaches consensus on the objectives, success criteria, constraints, and alternatives defined for the project and/or for the current cycle.

The project may modify the EoS as it reaches agreement. Document all changes with the change rationale in the meeting minutes and distribute the minutes to all attendees.

***Measurables*** The measurements to be taken for this activity are:

- The activity labor effort.
- The number and type of changes to the EoS.

## B.5 IDENTIFY RISKS



This activity is begun in quadrant two, Analyze and Avert Risks.

***Performers*** This activity is performed by the risk analyst, technical lead, and project manager.

***Inputs*** The inputs to this activity are:

- Approved Estimate of the Situation (EoS) and backup documents, including:
  - Contract.
  - Statement of Work.
  - Documents detailing the customer.
  - Internal infrastructure data.
- Master Plan documents.
- Project metrics and status reports.
- Configuration data.

***Outputs*** The output of this activity is a list of identified project and current cycle risk items documented in a draft risk analysis and aversion plan.

***Entrance Criteria*** The entrance criteria for this activity are:

- Project stakeholders have reviewed and approved the EoS document.
- The project has reached consensus to proceed with the current cycle.

***Exit Criteria*** The exit criteria for this activity are:

- The project has analyzed the project and current cycle objectives with respect to the alternatives and constraints and has identified the risk items (what can go wrong).
- The project has considered the impact of the possible unsatisfactory outcomes on the project and current cycle success criteria.

***Description***

In the Identify Risks activity the project comprehensively identifies potential project and/or current cycle risk items. Examine the objectives with respect to alternatives, constraints, organizational and project assets, and other process drivers, and identify what can go wrong. Examine these unsatisfactory outcomes and the effect on both the current cycle and project success criteria: that is, if a significant risk is identified for the current cycle, trace it back to the project-level objectives and success criteria to determine if the risk may impact the project as well as the cycle. To help identify typical project risks, use risk taxonomies such as in Boehm (1989, 117), U.S. Air Force (1988), and Charette (1990, 216-59).

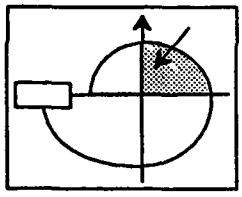
Document the results of this activity in a draft risk analysis and aversion plan. The plan is enhanced and matured as the remainder of the second quadrant activities are conducted.

***Measurables***

The measurements to be taken for this activity are:

- The total activity labor effort.
- The total number of project-level risk items.
- The total number of current cycle risk items.

## B.6 ANALYZE RISKS



This activity is begun in quadrant two, Analyze and Avert Risks.

***Performers*** This activity is performed by the risk analyst, technical lead, and project manager.

***Inputs*** The inputs to this activity are one or more identified project and cycle risk items documented in a draft risk analysis and aversion plan.

***Outputs*** The outputs of this activity are:

- An estimate of the probability and cost of occurrence for each of the risk items.
- Ranking of risk items.
- Updated draft risk analysis and aversion plan.

***Entrance Criteria*** The entrance criteria for this activity are:

- A list of identified project risks as complete as possible, given the current level of project understanding and knowledge.
- A list of identified current cycle risks as complete as possible, given the current level of cycle understanding and knowledge.

***Exit Criteria*** The exit criteria for this activity are:

- The project has estimated the likelihood of occurrence (probability of successful or unsuccessful outcome) for each risk item.
- The project has estimated the potential damage for each risk item (gain or cost associated with the outcome).
- The project has calculated an overall degree of risk.
- The project has stated explicitly any uncertainty in the estimates.

***Description*** To perform risk analysis, estimate the chance of potential loss (or gain) and the consequence (or benefit) of the risk situations previously identified. Analyze the risk situations independently of one another. Show any uncertainties in the estimates.

Charette (1989, 120-24) and the U.S. Air Force (1988) describe several approaches for developing measurement scales. Once risks are quantified,

compare, rank, and communicate them to appropriate stakeholders. Determine the high priority risks by identifying those with the highest calculated overall risk factor degrees based on the probability and cost of occurrence. It may be helpful to use a visual representation of risks as a communication and documentation tool. One risk visualization method, the iso-risk contour, is explained in Charette (1989).

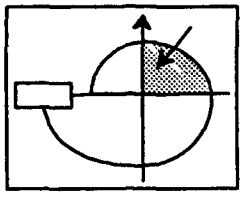
Document the results of this activity in the draft risk analysis and aversion plan. The plan is enhanced and matured as the remainder of the second quadrant activities are conducted.

***Measurables***

The measurements to be taken for this activity are:

- The total activity labor effort.
- The total number of high-priority risk items.

## B.7 EVALUATE RISKS



This activity is begun in quadrant two, Analyze and Avert Risks.

***Performers*** This activity is performed by the risk analyst, technical lead, and project manager.

***Inputs*** The inputs to this activity are:

- Draft risk analysis and aversion plan, including:
  - List of identified project and current cycle risk items.
  - List of high-priority risk items.
  - Probability of occurrence for each high-priority risk item.
  - Cost of occurrence for each high-priority risk item.
- Approved Estimate of the Situation (EoS) and backup documents, including:
  - Contract.
  - Statement of Work.
  - Documents describing the customer.
  - Internal infrastructure data.
- Master Plan documents.
- Project metrics and status reports.
- Engineering data.

***Outputs*** The outputs of this activity are risk aversion strategies for each high-priority risk item documented and included in the draft risk analysis and aversion plan.

***Entrance Criteria*** The entrance criterion for this activity is the list of high-priority project and current cycle risks, including the probability and cost of occurrence for each.

***Exit Criteria*** The exit criteria for this activity are:

- The project has elaborated, ranked, and analyzed the alternative strategies for averting the high-priority risks.
- The project has evaluated the interaction among the individual risks and has identified any additional risks or risk aggravation introduced as a result of each strategy.

- The project has documented the risk aversion strategies for each high-priority risk in the draft risk analysis and aversion plan, including any backup data or analysis as necessary to support the findings.
- The project has determined the amount of risk that is acceptable for each individual and overall project risk.

*Description*

During risk evaluation, identify risk aversion strategies and examine the impact of the risk aversion strategy on each high-priority risk item. Any risk aversion strategy identified should reduce the cost and/or probability of risk occurrence to an acceptable level. An option is to establish a risk referent, a measure against which to determine the amount of risk acceptable for each individual and overall project risk. Try to define a strategy that can reduce the risk to an acceptable level. In one case this may require a very active approach; in another it may mean acquiescing to the risk. Define the strategy activities to reduce the cost of occurrence of a risk and the probability of occurrence of a risk.

It is possible for risk aversion strategies to introduce new risks that will detract from the anticipated benefits or negatively impact other risks; therefore, address the impact of a risk aversion strategy on other risks and process drivers.

In general, consider the following when defining risk aversion strategies:

- Is the strategy feasible?
- Does the strategy reduce risk to an acceptable level?
- Will the strategy negatively impact another risk?
- What is the potential impact of new risks, if any, introduced by the strategy?
- Does the strategy support cycle and/or project objectives and success criteria?
- Are the tactics and means for implementing the strategy consistent with cycle and/or project constraints?
- Is the strategy cost-effective?

If applicable, the project can expand this activity to identify strategies that address high-priority process drivers other than those categorized as risks. Consider the draft risk analysis and aversion plan to be the draft process plan, since strategies for all drivers impacting the process are included in the document.

Use the *Using New Technologies: A Software Engineering Technology Transfer Guidebook* (Software Productivity Consortium 1992c) when identifying and evaluating strategies that involve new technology. The guidebook provides

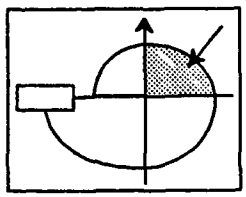
detailed guidance on identifying and selecting new technologies, and defines the process activities necessary to insert the technology for maximum use and benefit.

Document the results of this activity in the draft risk analysis and aversion plan. The plan is enhanced and matured as the remainder of the second quadrant activities are conducted.

***Measurables***

The measurement to be taken for this activity is the total activity labor effort.

## B.8 RISK REVIEW



This activity is begun in quadrant two, Analyze and Avert Risks.

***Performers*** This activity is performed by the project manager, risk analyst, technical lead, and engineer.

***Inputs*** The inputs to this activity are:

- Draft risk analysis and aversion plan.
- Approved Estimate of the Situation (EoS).

***Outputs*** The outputs of this activity are:

- Meeting minutes.
- Approval of the draft risk analysis and aversion plan.

***Entrance Criteria*** The entrance criteria for this activity are:

- The project has listed all cycle and project risk items as may be identified, given the current level of project and cycle understanding.
- The project has estimated the probability and cost of occurrence for each risk item.
- The project has determined the highest-priority risks, based on overall risk factors.
- The project has identified risk aversion strategies for each high-priority risk item and analyzed the impact of the strategy on other risks and process drivers.
- All project team members have received the draft risk analysis and aversion plan for review.

***Exit Criteria*** The exit criteria for this activity are:

- The project has reached consensus regarding any changes to the draft risk analysis and aversion plan.
- The project has documented all agreements or changes in the meeting minutes and distributed the minutes to all attendees.

***Description*** The Risk Review activity is an opportunity for the project team to review and comment on the results of the risk identification, analysis, and evaluation

activities. Submit the draft risk analysis and aversion plan to the rest of the team for review. A team meeting is a useful mechanism for discussing team comments and changes. The team should reach consensus on the identified risks, corresponding probability and cost of occurrence, high priority risk items, possible aversion strategies, and potential impact of aversion strategies.

If applicable, the project can expand this activity to review strategies that address high-priority process drivers other than those categorized as risks. Consider the draft risk analysis and aversion plan to be the draft process plan, since strategies for all drivers impacting the process are addressed in the document.

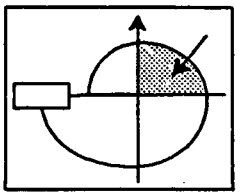
If the project modifies the draft risk analysis and aversion plan, document all changes with the change rationale in the meeting minutes and distribute the minutes to all attendees.

***Measurables***

The measurements to be taken for this activity are:

- The total activity labor effort.
- The number of changes to the risk analysis and aversion plan.
- The number of changes to the EoS.

## B.9 PLAN RISK AVERSION



This activity is begun in quadrant two, Analyze and Avert Risks.

***Performers***

This activity is performed by the project manager, risk analyst, technical lead, and engineer.

***Inputs***

The inputs to this activity are:

- Draft risk analysis and aversion plan, including:
  - List of identified project and current cycle risks.
  - Probability of occurrence for each risk item.
  - Cost of occurrence for each risk item.
  - Ranking of risk items.
  - Risk aversion strategies for each high-priority risk item.
- Approved Estimate of the Situation (EoS) and backup documents, including:
  - Contract.
  - Statement of Work.
  - Documents describing the customer.
  - Internal infrastructure data.
- Master Plan documents.
- Project metrics and status reports.
- Engineering data.

***Outputs***

The output of this activity is that the project has documented and included plans for the risk aversion strategies in the draft risk and aversion plan.

***Entrance Criteria***

The entrance criterion for this activity is that the project has obtained team review and consensus on the draft risk analysis and aversion plan to date.

***Exit Criteria***

The exit criteria for this activity are:

- The project has estimated the cost and schedule associated with each risk aversion strategy.
- The project has selected a recommended strategy for each risk item.
- The project has made recommendations to change the EoS by elaborating the alternatives, suggesting changes to the constraints, or defining new objectives.

***Description***

Estimate and evaluate the high-level cost and schedule for each risk item, and determine the best strategy for each. Try to plan risk aversion strategies so that their results are available in time to plan for the development activities that depend on them. Ideally the project team is involved in this activity to help evaluate and recommend the best risk aversion strategy(ies).

It is possible to manage and plan the risk reduction strategy through a new risk reduction cycle or subspiral. Make certain that the cycle or subspiral is planned so that any development which depends on the results is not started until after integration is complete.

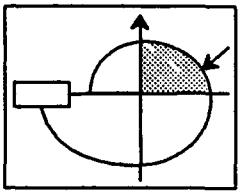
If applicable, the project can expand this activity to estimate the cost and schedule of strategies that address high-priority process drivers, other than those categorized as risks. Consider the draft risk analysis and aversion plan to be the draft process plan, since plans for all drivers impacting the process are included in the document.

Document the results of this activity in the draft risk analysis and aversion plan. The plan is enhanced and matured as the remainder of the second quadrant activities are conducted.

***Measurables***

The measurement to be taken for this activity is the total activity labor effort.

## B.10 COMMIT TO AVERSION PLAN



This activity is begun in quadrant two, Analyze and Avert Risks.

***Performers***

This activity is performed by all project stakeholders, including technical lead, engineer, contract manager, quality manager, user, project manager, risk analyst, and customer.

***Inputs***

The inputs to this activity are:

- Draft risk analysis and aversion plan, including:
  - List of identified project and current cycle risks.
  - Probability of occurrence for each risk item.
  - Cost of occurrence for each risk item.
  - Ranking of each risk item.
  - Risk aversion strategies for each high-priority risk item.
  - Cost and schedule for each risk aversion strategy.
  - Recommended strategies.
- Approved Estimate of the Situation (EoS).
- Master Plan documents.

***Outputs***

The outputs of this activity are:

- Meeting minutes.
- Approved risk analysis and aversion plan.
- Approved EoS.

***Entrance Criteria***

The entrance criteria for this activity are:

- The project has recommended the best candidate aversion strategy for each identified risk.
- The project has made recommendations to change the EoS by elaborating the alternatives, suggesting changes to the constraints, or defining new objectives.
- All individuals with an interest in the success of the project or their representatives have received the draft risk analysis and aversion plan for review.

***Exit Criteria***

The exit criteria for this activity are:

- Stakeholders have reached consensus on a risk aversion strategy for each high-priority risk item.
- Stakeholders have approved the risk analysis and aversion plan.
- Stakeholders have approved any changes to the EoS.

***Description***

The Commit to Aversion Plan activity is a mechanism for formally briefing all stakeholders on the contents of the risk analysis and aversion plan. As a result of the briefing, the project reaches consensus on and secures commitment to the risk aversion strategies recommended for each high-priority risk item. If consensus is not reached and commitment not secured, or if the risk aversion strategy committed to is not the one recommended, then the project may need to repeat the risk activities, as appropriate.

If applicable, the project can expand this activity to reach consensus on and secure commitment to strategies that address high-priority process drivers other than those categorized as risks. Consider the approved risk analysis and aversion plan to be the approved process plan, since strategies for all drivers impacting the process are addressed in the document.

The project must also reach consensus on any updates to the objectives, success criteria, constraints, and alternatives documented for the project and/or for the current cycle in the EoS.

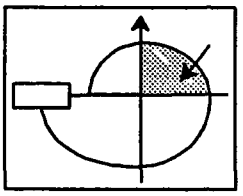
If the project modifies the draft risk analysis and aversion plan, or the EoS, document all changes with the change rationale in the meeting minutes and distribute the minutes to all attendees.

***Measurables***

The measurements to be taken for this activity are:

- The total activity labor effort.
- The number of changes to the risk analysis and aversion plan.
- The number of changes to the EoS.

## B.11 AVERT RISKS



This activity is begun in quadrant two, Analyze and Avert Risks.

***Performers*** This activity is performed by the project manager, risk analyst, technical lead, and engineer.

***Inputs*** The input to this activity is the approved risk analysis and aversion plan.

***Outputs*** The output of this activity is the development alternative for averting each high-priority risk item documented in an updated version of the risk analysis and aversion plan.

***Entrance Criteria*** The entrance criteria for this activity is that the project has reached consensus on a risk aversion strategy for each high-priority risk item.

***Exit Criteria*** The exit criteria for this activity are:

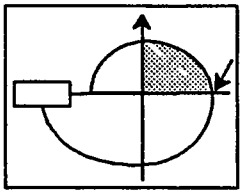
- The project has performed the tasks supporting the risk aversion strategy.
- The project has determined and justified a development alternative as a result of performing the risk aversion strategy.
- The project has assessed how the development strategy(ies) will affect the cycle and project process.
- The project has updated the risk analysis and aversion plan as appropriate.

***Description*** The Avert Risks activity performs tasks supporting the risk aversion strategy. These may include prototyping, simulation, surveys, comparative evaluations, evolutionary development, and other appropriate techniques. The project may spin off complicated or long-term risk aversion activities into their own spirals to be managed independently from main-line development. In performing the risk aversion strategy tasks, determine and justify a development alternative, such as a specific method, tool, or approach. Assess how this development alternative impacts the process for the cycle and for the remainder of the project. Document all relevant information by updating the risk analysis and aversion plan.

If applicable, the project can expand this activity to perform tasks that support strategies for high-priority process drivers, other than those categorized as risks. Consider the updated risk analysis and aversion plan to be an updated process plan, since the results of strategies for all drivers impacting the process are addressed in the document.

***Measurables*** The measurement to be taken for this activity is the total activity labor effort.

## B.12 COMMIT TO DEVELOPMENT ALTERNATIVE



This activity is begun in quadrant two, Analyze and Avert Risks.

***Performers***

This activity is performed by all project stakeholders, including technical lead, engineer, contract manager, quality manager, user, project manager, risk analyst, and customer.

***Inputs***

The inputs to this activity are:

- Updated risk analysis and aversion plan.
- Approved Estimate of the Situation (EoS).
- Master Plan documents.

***Outputs***

The outputs of this activity are:

- Meeting minutes.
- Approved risk analysis and aversion plan.

***Entrance Criteria***

The entrance criteria for this activity are:

- The project has determined and justified a development alternative as a result of performing each risk aversion strategy.
- All individuals with an interest in the success of the project or their representatives are participating.

***Exit Criteria***

The exit criteria for this activity are:

- The project has reached consensus on the development alternative(s) or approach(es) selected as a result of performing the risk aversion strategy(ies).
- The project has approved the updates to the risk analysis and aversion plan.

***Description***

The project presents the results of the Avert Risks activity to appropriate stakeholders and solicits commitment to the development alternative or approach selected as a result of performing the risk aversion strategy. If the development alternative selected as a result of the risk aversion activities is not committed to by senior management, then the project may need to repeat risk activities, as appropriate.

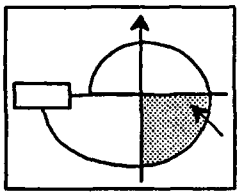
If applicable, the project can expand this activity to seek commitment to the development alternative or approaches to high-priority process drivers other than those categorized as risks. Consider the approved risk analysis and aversion plan to be the approved process plan, since development alternatives for all drivers impacting the process are included in the document.

***Measurables***

The measurements to be taken for this activity are:

- The total activity labor effort.
- The number of changes to the risk analysis and aversion plan.

## B.13 PLAN AND SCHEDULE



This activity is begun in quadrant three, Develop Product.

***Performers*** This activity is performed by the project manager, technical lead, risk analyst, and engineer.

***Inputs*** The inputs to this activity are:

- Approved risk analysis and aversion plan.
- Approved Estimate of the Situation (EoS).
- Master Plan documents.
- Project metrics and status reports.
- Engineering data.
- Internal infrastructure data.
- Documents describing the customer.

***Outputs*** The output of this activity is the draft cycle plan instantiated to an enactable level of detail.

***Entrance Criteria*** The entrance criteria for this activity are:

- The project has secured commitment to the development alternative(s).
- The project has collected and analyzed project status and engineering data from previous cycles and has updated the Master Plan documents accordingly.

***Exit Criteria*** The exit criteria for this activity are:

- The project has identified the cycle activities and documented them in the cycle plan.
  - The cycle activities are consistent with the objectives of the cycle and satisfy the cycle success criteria, as defined in the current version of the EoS.
  - The cycle activities are consistent with the development alternative(s) as documented in the current version of the risk analysis and aversion plan.

  - The cycle activities are consistent with the organizational process definition, if any or if appropriate.

- The project has defined each activity as consistent with the development alternative(s) committed to, and has tailored it from the organizational process definition, if any or if appropriate.

- The project has defined or tailored the project and engineering procedures as required to meet the cycle objectives and success criteria, accommodate cycle alternatives, satisfy cycle constraints, and avert cycle risks.

- The project has documented the dependencies between activities and the dependencies of the tasks within each activity.

- The project has estimated the costs and has opened work packages, or the lowest WBS level for the key cycle activities.

- The project has allocated resources to each activity.

- The project has defined intermediate milestones and performance measurements for each cycle activity.

- The project has scheduled the monitoring and review activities to accumulate data for and analyze the status of cycle activities.

- The project has scheduled sufficient interfacing events such as interface control working groups, management reviews, in-process reviews, etc., to coordinate this cycle's activities with other cycles.

***Description***

The main output of the Plan and Schedule activity is a detailed and enactable draft cycle plan. The project instantiates the Master Plan to address cycle objectives, success criteria, constraints, development alternatives, risks, and incorporate the legacy inherited from previous cycles.

In the Plan and Schedule activity the project identifies and defines the technical tasks to be accomplished within each activity to achieve the cycle objectives and specifies what methods, practices, or tools to be used to complete each task. Tailor the project and engineering procedures accordingly. Include in the activities sufficient interfacing events (such as interface control working groups, management reviews, in-process reviews) to coordinate this cycle's activities with other cycles, as well as satisfy customer control and monitoring needs. In addition:

- Sequence the identified technical tasks and activities.

- Estimate the total labor effort for each planned and scheduled cycle activity.

- Define work packages for the key activities defined for the current cycle.
- Estimate the number and type of product components (e.g., modules, documentation) to be produced as a result of enacting the cycle plan.
- Estimate the size of each product component.
- Estimate the number, severity, and source of errors likely to be found during verification activities.

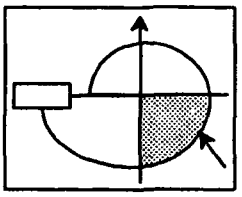
During this activity, remember to:

- Address any customer requirements.
- Comply with customer policies, procedures, standards, and regulations, if necessary.
- Comply with the project's organizational process definition, if available and appropriate.
- Use any available historical planning and engineering data for estimating purposes.
- Take advantage of in-house methods, tools, training, and support.

***Measurables***

The measurement to be taken for this activity is the total activity labor effort.

## B.14 COMMIT TO PLAN



This activity is begun in quadrant three, Develop Product.

***Performers***

This activity is performed by the project manager, risk analyst, technical lead, and engineer.

***Inputs***

The inputs to this activity are:

- Draft cycle plan.
- Approved Estimate of the Situation.

***Outputs***

The outputs of this activity are:

- Meeting minutes.
- Approved cycle plan/enactable cycle process.

***Entrance Criteria***

The entrance criterion for this activity is that the project has a draft cycle plan instantiated to an enactable level of detail.

***Exit Criteria***

The exit criteria for this activity are:

- The project has reached consensus regarding any changes to the draft cycle plan.
- The project has documented all agreements or changes in the meeting minutes and distributed the minutes to all attendees.
- The project has reached a consensus to proceed with the cycle by enacting the cycle plan.
- The project has opened work packages.

***Description***

The Commit to Plan activity is an opportunity for a project team to review and comment on the results of the cycle planning and scheduling activities.The project must reach consensus that the activities of the cycle plan are appropriate to meeting cycle objectives. The cycle plan may be updated according to team comments.

Once the project has secured commitment to the detailed cycle plan, the project may decide to formally brief senior management on the direction the current cycle development activities will take. This briefing may be a natural extension of any existing, periodic internal management reviews, or may need

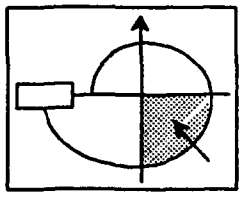
to be scheduled separately. As a result of committing to the cycle plan, work packages are opened and assigned, and the detailed cycle process, as documented in the cycle plan, is considered enactable.

***Measurables***

The measurements to be taken for this activity are:

- The total activity labor effort.
- The number of changes to the cycle plan.

## B.15 DEVELOP AND VERIFY PRODUCT



This activity is begun in quadrant three, Develop Product.

***Performers***

This activity is performed by the user, project manager, risk analyst, technical lead, engineer, and customer.

***Inputs***

The inputs to this activity in the **first cycle** are:

- Approved risk analysis and aversion plan.
- Approved Estimate of the Situation (EoS) and supporting documents.
- Internal infrastructure data.
- Documents describing the customer.

The inputs to this activity in the **first cycle** as well as **subsequent cycles** are:

- Approved cycle plan/enactable cycle process.
- Software and documentation baselines produced in previous cycles (if applicable).

***Outputs***

The outputs of this activity in the **first cycle** are:

- Partial Master Plan, including:
  - Project-level risk analysis and aversion plan.
  - Project-level EoS.
  - Process model for the spiral.
  - Definitions of deliverable.
  - WBS and budget allocated to cost account level.
  - Project-level schedules, project organization, and committed resources.
  - Software configuration management plans.
  - Other required planning information.
- Project-level process definition (subset of the Master Plan).

The outputs of this activity in the **first cycle** as well as **subsequent cycles** are:

- Product or part of the product necessary to meet cycle objectives and satisfy cycle success criteria.
- Enacted cycle process.

***Entrance Criteria***

The entrance criterion for this activity in the **first cycle** is that the project has an approved cycle plan instantiated to an enactable level of detail for the first cycle, i.e., a "plan the plan."

The entrance criterion for this activity in **subsequent cycles** is that the project has an approved cycle plan instantiated to an enactable level of detail for the current cycle.

***Exit Criteria***

The exit criteria of this activity in the **first cycle** are:

- The project has completely enacted the cycle process as defined in the cycle plan for the first cycle.
- The project has chosen and documented a process model for the spiral.
- The project has documented all the products and services to be provided as a result of the project.
- The work to be performed is consistent with the requirements of the contract.
- The level of detail in the plan is commensurate with the level of current understanding about the project and its process drivers.
- The project has defined an organization and assigned responsibility.
- All persons affected by the project plans or their designated representatives have verified and made a commitment to those plans.

The exit criteria of this activity in **subsequent cycles** are:

- The project has completely enacted the cycle process as defined in the cycle plan.
- The project has built the product or part of the product necessary to meet cycle objectives and satisfy cycle success criteria.
- The project has verified that product components satisfy all allocated requirements.
- The project has verified that product components are correct and consistent with the design, if applicable.
- The project has verified that product components are maintainable and understandable.

- The project has verified that product components comply with project standards.

***Description***

In the Define Project/Cycle activity in the **first cycle**, the project defines the specific work to be accomplished to deliver the desired products and services and documents the project approach in the project-level Master Plan. The project considers the current status of the project, as reflected by the results of the activities performed to date, the existing engineering data, and the current concept of operations. Define the framework to be used to guide the project. Specify the deliverable products and the top-level project milestones by which the project will measure progress. In this activity, the project:

- Establishes the overall strategy to be used to address the project. This is the process model for the project spiral(s).
- Defines each major project deliverable (e.g., "builds," data items) and the services to be provided.
- Determines the high-level activities that the project must accomplish to develop and deliver each product and service in accordance with the development strategy.
- Ensures that the work to be performed is consistent with any contract requirements, such as a Statement of Work, and any project-specific risks determined by the project risk assessment.
- Identifies any project constraints such as cost, schedule, quality, and technical performance.
- Defines the organizational context for the work, assigning responsibility.
- Prepares draft cycle definitions for the project cycles the project anticipates.

The project instantiates and evolves the detail in the Master Plan for each cycle, based on cycle process drivers, strategies, and alternatives.

In **subsequent cycles**, the project develops the product or part of the product, inspects the results for technical merit, and verifies results against cycle objectives and success criteria. The project enacts the specific activities in the third quadrant identified as a result of executing the first two quadrants, and the planning and scheduling activity during the first part of the third quadrant. That is, the project enacts main-line, or traditional, software development activities as scheduled and planned for the current cycle, such as requirements, design, code, integration, and test. The project performs verification as an integral part of the third quadrant development activities. Verification ensures that the artifacts or advanced product maturity produced by the development activities meet quality requirements, cycle objectives, and success criteria.

Verification techniques may be informal walkthroughs or reviews, or formal inspections.

The project verifies that the artifacts or advanced product maturity produced by the development activities meet quality requirements, cycle objectives, and success criteria. Capture and record:

- Product error data.
- Product change data.
- Configuration identification.

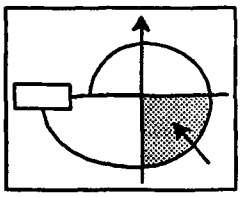
***Measurables***

The measurements to be taken for this activity are:

- Actual total labor effort for each planned and scheduled cycle activity.
- Actual number and type of product components (e.g., modules, documentation) produced.
- Actual size of each product component.
- Actual number, severity, and source of errors found during verification activities.
- Actual risk increase or decrease of each high-priority risk.
- Overall project risk.

The *Software Measurement Guidebook* (Software Productivity Consortium 1992b) is a good reference source for more specific guidance on defining and collecting size, cost, schedule, and error measurements.

## B.16 MONITOR AND REVIEW



This activity is begun in quadrant three, Develop Product.

***Performers*** This activity is performed by the project manager, risk analyst, technical lead, and engineer.

***Inputs*** The inputs to this activity are:

- Approved cycle plan/enactable cycle process.
- Approved risk analysis and aversion plan.
- Approved Estimate of the Situation and supporting documents.
- Master Plan documents.
- Cycle measurables and status.

***Outputs*** The outputs of this activity are:

- Cycle status reports.
- Updated cycle plan/evolved cycle process.
- Meeting minutes.

***Entrance Criteria*** The entrance criterion for this activity is that the project is enacting the cycle process as defined in the cycle plan.

***Exit Criteria*** The exit criteria for this activity are:

- The project has collected and reviewed cycle data.
- The project has produced and distributed status reports.

***Description*** The Monitor and Review activity maintains management control over the development process. The project performs this activity in parallel with the Develop and Verify Product activity. Periodically capture and analyze the status of the cycle, which provides management with insight into the cost, schedule, and technical performance status of the cycle. This activity takes the raw activity progress status and analyzes it to produce management metrics that support decision making regarding corrective action. Collect and analyze the work package status in each cost account and summarize the status to the highest schedule level. For each parameter selected for monitoring, update the actual and projected performance, analyze the trends, and note unfavorable

trends or values. Specifically, the project reviews and analyzes the following based on the data collected during the Develop and Verify Product activity:

- Product and component size status.
- Product and component error data.
- Product and component change data.
- Cycle schedule status.
- Cycle cost status.
- Cycle risk status.
- Cycle quality status.
- Cycle organizational status.

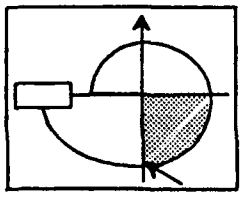
The *Software Measurement Guidebook* (Software Productivity Consortium 1992b) is a good reference source for specific size, cost, schedule, and error measurement and analysis methods.

The project conducts periodic cycle reviews. These reviews serve two purposes. First, they demonstrate to the project team and other stakeholders that the current cycle and overall project are under control. Second, they provide a mechanism to obtain a commitment to proceed with the cycle plan. At these reviews a project can make decisions to reallocate resources, replan schedules, or reassess risks for the current cycle activities. All such changes are reflected in an updated cycle plan/evolved cycle process.

***Measurables***

The measurement to be taken for this activity is the total activity labor effort.

## B.17 TECHNICAL PRODUCT REVIEW



This activity is begun in quadrant three, Develop Product.

***Performers*** This activity is performed by the project manager, technical lead, and engineer.

***Inputs*** The inputs to this activity are:

- Product or part of the product produced as a result of enacting the cycle process.
- Enacted cycle process as defined in the updated cycle plan.
- Approved risk analysis and aversion plan.
- Approved Estimate of the Situation and supporting documents.
- Master Plan documents.
- Project metrics and status reports.

***Outputs*** The output of this activity is a product or part of the product that meets cycle objectives, success criteria, and quality assurance.

***Entrance Criteria*** The entrance criterion for this activity is that the project has enacted the cycle process as defined in the cycle plan and has produced a product or part of the product.

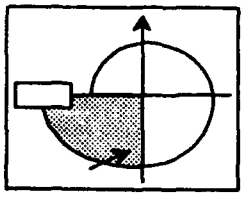
***Exit Criteria*** The exit criteria for this activity are:

- The project has reviewed the product or part of the product to ensure that cycle objectives and success criteria were met.
- The project has reviewed the results of verification activities to assure the technical quality of the product or part of the product produced.

***Description*** The technical product review is a project team review of the product or part of the product developed to ensure that cycle objectives and success criteria were met. This review is an opportunity for project review; however, the ongoing Monitor and Review activity should have identified and resolved any factor that might have had a negative impact on meeting cycle objectives and/or success criteria.

***Measurables*** The measurement to be taken for this activity is the total activity labor effort.

## B.18 BASELINE PRODUCT



This activity is begun in quadrant four, Manage and Plan.

***Performers*** This activity is performed by the configuration manager.

***Inputs*** The inputs to this activity are:

- Product or part of the product that meets cycle objectives, success criteria, and quality assurance.
- Software configuration management plan.
- Product error data.
- Product change data.
- Configuration identification.

***Outputs*** The outputs of this activity are:

- Software and documentation baselines.
- Configuration status.
- Metrics.

***Entrance Criteria*** The entrance criteria for this activity are:

- A software CM plan is included as part of the Master Plan.
- The project has developed a product or part of the product as a result of enacting the cycle process and needs to place it under configuration control.
- The project has reviewed product error and change data.
- The project has received a baselined configuration identification.

***Exit Criteria*** The exit criterion for this activity is that the project has established the software and/or documentation baseline.

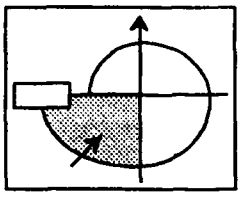
***Description*** Once cycle objectives and success criteria are met and the product developed in the third quadrant is approved by the team during the technical product review, the project baselines the product or part of the product produced during the cycle. Track the implementation of changes to controlled software products to ensure that the configuration of the product is evident at all times.

Establish each baseline and track all subsequent changes in the configuration status. Maintain the history of changes to each configuration item throughout the software life cycle for status accounting. Store all items under control in a secure, limited-access software development library.

***Measurables***

The measurement to be taken for this activity is the total activity labor effort.

## B.19 REVIEW PROGRESS



This activity is begun in quadrant four, Manage and Plan.

***Performers*** This activity is performed by the technical lead, engineer, contract manager, quality manager, project manager, and risk analyst.

***Inputs*** The inputs to this activity are:

- Enacted cycle process as defined in the updated cycle plan.
- Cycle status reports.
- Approved risk analysis and aversion plan.
- Approved Estimate of the Situation and supporting documents.
- Master Plan documents.
- Product or part of the product that meets cycle objectives, success criteria, and quality assurance.
- Software configuration management plan.
- Product error data.
- Product change data.
- Configuration identification.

***Outputs*** The outputs of this activity are:

- Cycle and project metrics.
- Updated project process drivers.

***Entrance Criteria*** The entrance criteria for this activity are:

- The cycle process as defined in the cycle plan has been enacted.
- A baselined product exists.

***Exit Criteria*** The exit criteria for this activity are:

- The project has reviewed final cycle status.
- The project has produced actual-versus-estimate metrics for both cycle and project.

- The project has compiled any lessons learned from the cycle.
- The project has identified changes to project process drivers, based on the enacted cycle process and resulting cycle product, data, and status.

***Description***

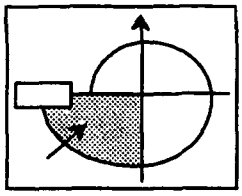
In the Review Progress activity the project evaluates cycle plan actual measures versus those estimated, success criteria, and lessons learned. Analyze how the final cycle status may impact the process for the rest of the project. Specifically, review cycle data and produce the following actual-versus-estimate metrics:

***Measurables***

The measurements to be taken for this activity are:

- Number and type of product components.
- Component size.
- Activity, cycle, and overall project schedule.
- Activity, cycle, and overall project labor hours and dollars.
- Component and product errors.

## B.20 DEVELOP/UPDATE ENGINEERING AND PROJECT PROCEDURES



This activity is begun in quadrant four, Manage and Plan.

***Performers***

This activity is performed by the technical lead, engineer, quality manager, project manager, and risk analyst.

***Inputs***

The inputs to this activity in the **first cycle** are:

- Partial Master Plan, including:
  - Project-level risk analysis and aversion plan.
  - Project-level Estimate of the Situation.
  - Process model for the spiral.
  - Deliverable definitions.
  - WBS and budget allocated to cost account level.
  - Project-level schedules, project organization charts, and committed resources.
  - Software configuration management plans.
  - Other required planning information.
- Internal infrastructure data.
- Documents describing the customer.

The inputs of this activity in the **first cycle** and **subsequent cycles** are:

- Enacted cycle process as defined in the updated cycle plan.
- Cycle status reports.
- Cycle and project metrics.
- Cycle and project lessons learned.
- Updated project process drivers.

***Outputs***

The output of this activity in the **first cycle** is the completed Master Plan.

The outputs of this activity in **subsequent cycles** are updated engineering and project procedures.

*Entrance Criteria*

The entrance criteria for this activity in the **first cycle** are:

- The project has baselined existing Master Plan documents.
- The project has collected all organizational and customer information and data.

The entrance criteria for this activity in the **first cycle** and **subsequent cycles** are:

- The project has updated project status to reflect cycle status.
- The project has updated project process drivers.
- The project has compiled lessons learned to date.

*Exit Criteria*

The exit criterion for this activity is that the project has developed or updated the engineering and project procedures.

*Description*

The project develops and updates engineering and project procedures, including activity and artifact specifications and method descriptions. Project engineering and project procedures should:

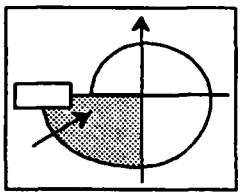
- Specify activities in a consistent way, either by using a standard format or a specific notation.
- Specify any artifacts produced by the activity.
- Describe how to perform or use the development methods/tools/practices alternative selected to support the activity.
- Specify measurables, if any, that should be collected when performing the activity.

Define procedures by tailoring the organizational process definition, if one exists. The project may also need to incorporate client requirements to follow certain standards or development alternatives. Define the engineering and project procedures only to the level of detail appropriate to the current understanding of the project. For example, if the project does not yet know what design method will be used, then define the design procedure at an outline level to be updated in a later cycle. Once defined, the project can update and evolve the engineering and project procedures as necessary, based on lessons learned, updates to the process drivers, or a new or changed development alternative.

*Measurables*

The measurement to be taken for this activity is the total activity labor effort.

## B.21 UPDATE MASTER PLAN



This activity is begun in quadrant four, Manage and Plan.

***Performers*** This activity is performed by the project manager, risk analyst, technical lead, corporate manager, contract manager, and quality manager.

***Inputs*** The inputs to this activity are:

- Enacted cycle process as defined in the updated cycle plan.
- Cycle status reports.
- Cycle and project metrics.
- Cycle and project lessons learned.
- Updated project process drivers.
- Configuration data.
- Updated engineering and project procedures.

***Outputs*** The output of this activity is the updated Master Plan.

***Entrance Criteria*** The entrance criterion for this activity is that the project has completed all other activities in this cycle.

***Exit Criteria*** The exit criterion for this activity is that the project has updated the Master Plan for the project.

***Description*** In the Update Master Plan activity, the project evolves the remainder of the project process as defined in the Master Plan to reflect enacted cycle process and metrics. Analyze the lessons learned, project and cycle status reports and metrics, and updated process drivers. Update the project process, as defined in the Master Plan documents, with the following information:

- Updated engineering and project procedures.
- High-level schedule for remainder of project.
- Cost in dollars and labor hours for remainder of project.
- Number of key project risks remaining.
- Risk increase or decrease of each high-priority risk and overall project risk.

- Errors identified, solved, and remaining.
- Organizational status.
- Lessons learned.

Although much of this information may have been gathered and analyzed in previous activities, this activity gives the project the opportunity to examine thoroughly the impact of each part of the updated process on the rest of the process. For example, compose and analyze the answers to questions such as:

- How do the lessons learned impact the project schedule?
- How do the updated engineering and project procedures affect risk increase or reduction of each high-priority risk?
- Should the labor hour estimates for the remainder of the project be updated as a result of the lessons learned?

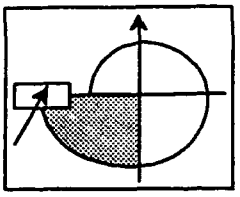
Update the Master Plan documents to the level of detail appropriate to the current level of understanding of the project and its process drivers.

The *Software Measurement Guidebook* (Software Productivity Consortium 1992b) is a good reference source for specific size, cost, schedule, and error measurement and analysis methods.

***Measurables***

The measurement to be taken for this activity is the total activity labor effort.

## B.22 MANAGEMENT REVIEW AND COMMITMENT TO PURSUE PROJECT



This activity is begun in quadrant four, Manage and Plan.

***Performers*** This activity is performed by all project stakeholders, including: technical lead, engineer, contract manager, quality manager, user, project manager, risk analyst, and customer.

***Inputs*** The input to this activity is the updated Master Plan.

***Outputs*** The outputs of this activity are:

- Meeting minutes.
- Approved Master Plan.

***Entrance Criteria*** The entrance criteria for this activity are:

- The Master Plan document is drafted for the project or updated following an enacted cycle.
- All individuals with an interest in the success of the project or their representatives are participating.

***Exit Criteria*** The exit criteria for this activity are:

- The project has reached a consensus to proceed with the project as a result of reviewing the Master Plan.
- The project has reached consensus on the objectives and success criteria for the next cycle.
- The project has documented all agreements and/or changes in the meeting minutes and distributed the minutes to all attendees.

***Description*** The commit concept, which comes into play at the end of the fourth quadrant, is one of the most important in using the spiral model. It is similar in purpose to a baseline. All stakeholders in a project should be briefed on the results of the current cycle and on any changes in plans, and should agree with the decisions made regarding what to do next. Stakeholders include project management at all levels, as well as the project team. The review may also include customer representatives. The purpose of the review is for a project to determine whether project-level objectives, alternatives, and constraints are still feasible and on-track, agree on the objectives and success criteria for the next cycle, and commit resources to that cycle.

The project may decide to modify the Master Plan as the stakeholders reach agreement. Document all changes with the change rationale in the meeting minutes and distribute the minutes to all attendees.

***Measurables***

The measurements to be taken for this activity are:

- The activity labor effort.
- The number and type of changes to the Master Plan.

*This page intentionally left blank.*

# GLOSSARY

| | |
|---|---|
| Artifact | The physical representation of an activity input or output, such as a development plan or design document. |
| Business area | A group of projects supporting a similar product line or serving a similar customer base. |
| Constraint | An externally-imposed and -controlled limit. |
| Cycle | A traversal of all four quadrants of the spiral model, which denotes that the product has matured by a specified amount. |
| Cycle process | The enactable process for the current cycle. |
| Cycle process driver | The key characteristics that affect the instantiation of the process for the current cycle. Cycle process drivers are generally a decomposition of project process drivers or are inherited from the previous cycles. |
| Development alternative | A specific method, strategy, or approach for performing an activity. |
| Domain | A coherent business-area organization whose mission is the production and delivery of similar products, or which serves a similar customer base. |
| Enact | To perform an activity in the development process. |
| Entrance criteria | The requirements that must be met before an activity can be started. |
| Estimate of the Situation (EoS) | A document that identifies the project's goals, strategies, product and process assumptions, and the assets available for performing the project. |
| Evolutionary development | A system development approach in which software is incrementally developed and refined through customer use and resulting feedback between users and the development team. |
| Evolve | To change an organizational or project process definition in order to correct problems or to address newly discovered or better-understood process drivers. |
| Exit criteria | The requirements that must be met before an activity can be completed. |

| | |
|---|---|
| Incremental development | A system development approach in which functional increments are built and delivered. The definitions for all increments are fixed at the beginning of the project. |
| Instantiate | To create an enactable process from an existing process definition, generally based on the selected alternatives for addressing the relevant process drivers. |
| Master Plan | A plan produced during the first cycle of a spiral describing a detailed plan for executing the first cycle and a high-level plan for the remainder of the spiral. |
| Method | A task, rule, guideline, or technique that describes how to perform an activity. |
| Notation | A method for specifying activities in a consistent way. A notation may be an informal natural language description of an activity or a formal specification language. |
| Objective | The intended result of a course of action. |
| Organizational process driver | A key characteristic of an organization that affects an organizational process definition. |
| Process | A set of actions intended to reach a goal. |
| Process automation | The use of a machine to automate all or part of the software development process. |
| Process definition | An artifact defining a software development process. A process definition may be at a high-level process model or may include all the information necessary to enact the process. |
| Process driver | A key characteristic that places requirements on the definition or instantiation of the software development process. |
| Process engineering | Activities that attempt to ensure that a process definition is the best one, given an organization's or project's unique process drivers. |
| Process evaluation | Activities that analyze process measurements and recommend process improvements. |
| Process model | An abstract representation of all or part of a process used when the use of the complete process is not desirable or practical. |

| | |
|---|---|
| Project process driver | A key characteristic that impacts process definition at the project level. Process drivers include the organizational process definition, project objectives, project constraints, and project risks. |
| Resource | Personnel or equipment that can be used to perform an activity. |
| Risk | A potential problem area that may affect the successful completion of a project. |
| Risk management | A management approach focused on identifying and removing risks before they become threats to successful project completion. |
| Risk referent | A measure of acceptable risk for each individual and overall project risk. |
| Software development process | A set of activities to produce software and supporting products. |
| Spiral | One or more cycles that, when completed, create a specific accomplishment such as a deliverable or artifact. |
| Spiral model | A process model that provides a disciplined development framework while eliminating the lock-step, linear progression of stages characteristic of many other software development models. |
| Stakeholder | An individual with a vested interest in a project, such as a team member, client, or manager. |
| Success criteria | The minimum acceptable requirements that must be accomplished in the cycle for the project to make progress. |
| Synthesis process model | A process model that supports construction of software systems as a group of systems that have similar descriptions. |
| Tailor | Modify an organizational process definition for a specific project, based on that project's process drivers. |
| Task | A unit of work in an activity. |
| User | A person who works with a product after it has been delivered. |
| Validation | Tracking the quality and completeness of an activity. |
| Waterfall process model | A process model that provides a linear progression of software development stages. |

| | |
|---|---|
| Work breakdown structure (WBS) | A description of work to be done in terms of a hierarchy of short, manageable tasks with specific inputs, outputs, schedules, and assigned responsibilities. |
| Work package | The lowest work breakdown structure level. |

# REFERENCES


Ashley, David B. 1990 — Project Risk Identification Using Inference Subjective Expert Assessment and Historical Data. In *The State of the Art in Project Risk Management, Proceedings of the INTERNET International Expert Seminar in connection with the PMI/INTERNET Joint Symposium, Atlanta, October 12–13, 1989.* Zurich, Switzerland: International Project Management Association, 9–25.

Boehm, Barry W. 1986 — A Spiral Model of Software Development and Enhancement. *ACM Software Engineering Notes* 11:22-42.

1988 — A Spiral Model of Software Development and Enhancement. *IEEE Computer* 21:61-72.

1989 — *Tutorial: Software Risk Management*. Washington, D.C.: IEEE Computer Society Press.

Boehm, Barry W. and Frank Belz 1988 — Experiences with the Spiral Model as a Process Model Generation. *Proceedings of the 4th International Software Process Workshop.*

Charette, Robert N. 1989 — *Software Engineering Risk Analysis and Management*. New York, New York: Intertext Publications, McGraw-Hill.

1990 — *Applications Strategies for Risk Analysis*. New York, New York: Intertext Publications, McGraw-Hill.

1991 — Risk Management Seminar. Herndon, Virginia: Software Productivity Consortium.

Department of Defense 1988 — *Military Standard: Defense System Software Development,* (DOD-STD-2167A). Washington, D.C.: Department of Defense.

1991 — *Defense Acquisition Management Policies and Procedures,* Department of Defense Instruction 5000.2. Washington, D.C.: Department of Defense.

Feiler, Peter H. and Watts S. Humphrey 1992 — *Software Process Development and Enactment: Concepts and Definitions*, CMU/SEI-92-TR-4, DRAFT. Pittsburgh, Pennsylvania: Carnegie-Mellon University, Software Engineering Institute.

Gilbreath, Robert 1986 — *Winning at Project Management*. New York, New York: John Wiley & Sons.

IEEE Computer Society
1992

*IEEE Standard for Developing Life Cycle Processes* (IEEE STD 1974). New York, New York: Institute of Electrical and Electronics Engineers, Inc.

Marca, David A., and
Clement L. McGowan
1987

*SADT™ Structured Analysis and Design Technique.* New York, New York: McGraw-Hill.

Odiorne, George S.
1979

*MBO II: A System of Managerial Leadership for the 80s.* Belmont, California: Fearson Pitman Publishers.

Ould, Martyn
1990

*Strategies for Software Engineering: Management of Risk and Quality.* Chichester, England: John Wiley & Sons.

Over, James W.
1991

STARS '91 Process Asset Library. *Proceedings, STARS '91, December 3-4, 1991.*

Paulk, Mark C., Bill Curtis,
Mary Beth Chrissis,
Edward L. Averill,
Judy Bamberger,
Timothy C. Kasse,
Mike Konrad, Jeffrey R. Perdue,
Charles V. Weber, and
James V. Withey
1991

*Capability Maturity Model for Software,* CMU/SEI-91-TR-24 (also published as ESD-TR-91-24). Pittsburgh, Pennsylvania: Software Engineering Institute, Carnegie Mellon University.

Radice, Ronald A., and
Richard W. Phillips
1988

*Software Engineering: An Industrial Approach,* Vol. 1. Englewood Cliffs, New Jersey: Prentice-Hall.

Royce, Winston W.
1970

Managing the Development of Large Software Systems. *Proceedings, IEEE WESCON.*

Snyder, James C. and
Anthony J. Catanese
1979

*Introduction to Architecture.* New York, New York: McGraw-Hill.

Software Productivity
Consortium
1991

*Synthesis Guidebook,* SPC-91122-MC. Herndon, Virginia: Software Productivity Consortium.

1992a

*Process Definition and Modeling Guidebook,* SPC-92041-CMC. Herndon, Virginia: Software Productivity Consortium.

1992b

*Software Measurement Guidebook,* SPC-91060-CMC. Herndon, Virginia: Software Productivity Consortium.

1992c — *Using New Technologies: A Software Engineering Technology Transfer Guidebook*, SPC-92046-CMC. Herndon, Virginia: Software Productivity Consortium.

U.S. Air Force 1988 — *Acquisition Management: Software Risk Abatement*, AFSC/AFLCP 800–45. Air Force Systems Command and Air Force Logistics Command.

Weber, Charles V., Mark C. Paulk, Cynthia J. Wise, James V. Withey, edited by Mary Beth Chrissis, Suzanne D. Couturiaux, and Ginny Redish 1991 — *Key Practices for the Capability Maturity Model*, CMU/SEI-91-TR-25 (also published as ESD-TR-91-25). Pittsburgh, Pennsylvania: Software Engineering Institute, Carnegie Mellon University.

Wild, Chris, Kurt Maley, and Linfang Liu 1991 — Decision-Based Software Development. *Software Maintenance: Research and Practice* 3:17–43.

*This page intentionally left blank.*